Temporal Mode-Checking for Runtime Monitoring of Privacy Policies

Omar Chowdhury, Limin Jia, Deepak Garg, Anupam Datta

May 28, 2014

CMU-CyLab-14-005

CyLab
Carnegie Mellon University
Pittsburgh, PA 15213

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **28 MAY 2014**	2. REPORT TYPE	3. DATES COVERED **00-00-2014 to 00-00-2014**

4. TITLE AND SUBTITLE **Temporal Mode-Checking for Runtime Monitoring of Privacy Policies**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,CyLab,Pittsburgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Fragments of first-order temporal logic are useful for representing many practical privacy and security policies. Past work has proposed two strategies for checking event trace (audit log) compliance with policies: online monitoring and offline audit. Although online monitoring is space- and timeefficient, existing techniques insist that satisfying instances of all subformulas of the policy be amenable to caching, which limits expressiveness when some subformulas have infinite support. In contrast, offline audit is brute force and can handle more policies but is not as efficient. This paper proposes a new online monitoring algorithm that caches satisfying instances when it can and falls back to the brute force search when it cannot. Our key technical insight is a new flowand time-sensitive static check of variable groundedness, called the temporal mode check, which determines subformulas for which such caching is feasible and those for which it is not and, hence guides our algorithm. We prove the correctness of our algorithm and evaluate its performance over synthetic traces and realistic policies.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**73**	

Temporal Mode-Checking for Runtime Monitoring of Privacy Policies

Omar Chowdhury **Limin Jia** **Deepak Garg**
Anupam Datta

May 2014
CMU-CyLab-14-005

CyLab
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Fragments of first-order temporal logic are useful for representing many practical privacy and security policies. Past work has proposed two strategies for checking event trace (audit log) compliance with policies: online monitoring and offline audit. Although online monitoring is space- and time-efficient, existing techniques insist that satisfying instances of all subformulas of the policy be amenable to caching, which limits expressiveness when some subformulas have infinite support. In contrast, offline audit is brute force and can handle more policies but is not as efficient. This paper proposes a new online monitoring algorithm that caches satisfying instances when it can, and falls back to the brute force search when it cannot. Our key technical insight is a new flow- and time-sensitive static check of variable groundedness, called the *temporal mode check*, which determines subformulas for which such caching is feasible and those for which it is not and, hence, guides our algorithm. We prove the correctness of our algorithm and evaluate its performance over synthetic traces and realistic policies.

✠ This is the extended version of the paper titled "*Temporal Mode-Checking for Runtime Monitoring of Privacy Policies*" that appears in the 26th International Conference on Computer Aided Verification (CAV) 2014. All the opinions expressed in this paper represent only the authors' views.

Keywords: Mode checking, runtime monitoring, metric first-order temporal logic, privacy policy, privacy legislation, privacy policy compliance checking, HIPAA, GLBA.

1 Introduction

Many organizations routinely collect sensitive personal information like medical and financial records to carry out business operations and to provide services to clients. These organizations must handle sensitive information in compliance with applicable privacy legislation like the Health Insurance Portability and Accountability Act (HIPAA) [1] and the Gramm-Leach-Bliley Act (GLBA) [2]. Violations attract substantial monetary and even criminal penalties [3]. Hence, developing mechanisms and automatic tools to check privacy policy compliance in organizations is an important problem.

The overarching goal of this paper is to improve the state of the art in checking whether an event trace or audit log, which records relevant events of an organization's data handling operations, is compliant with a given privacy policy. At a high-level, this problem can be approached in two different ways. First, logs may be recorded and compliance may be checked *offline*, when demanded by an audit authority. Alternatively, an *online* program may monitor privacy-relevant events, check them against the prevailing privacy policy and report violations on the fly. Both approaches have been considered in literature: An algorithm for offline compliance checking has been proposed by a subset of the authors [4], whereas online monitoring has been the subject of extensive work by other researchers [5–11].

These two lines of work have two common features. First, they both assume that privacy policies are represented in first-order temporal logic, extended with explicit time. Such extensions have been demonstrated adequate for representing the privacy requirements of both HIPAA and GLBA [12]. Second, to ensure that only finitely many instances of quantifiers are tested during compliance checking, both lines of work use static policy checks to restrict the syntax of the logic. The specific static checks vary, but always rely on assumptions about finiteness of predicates provided by the policy designer. Some work, e.g. [5, 8–11], is based on the *safe-range check* [5], which requires syntactic subformulas to have finite support independent of each other; other work, e.g. [4, 7], is based on the *mode check* from logic programming [13–15], which is more general and can propagate variable groundedness information across subformulas.

Both lines of work have their relative advantages and disadvantages. An online monitor can cache policy-relevant information from logs on the fly (in so-called *summary structures*) and discard the remaining log immediately. This saves space. It also saves time because the summary structures are organized according to the policy formula so lookups are quicker than scans of the log in the offline method. However, online monitoring algorithms proposed so far require that all subformulas of the policy formula be amenable to caching. Furthermore, many real policies, including several privacy requirements of HIPAA and GLBA, are not amenable to such caching. In contrast, the offline algorithm proposed in our prior work [4] uses brute force search over a stored log. This is inefficient when compared to an online monitor, but it can handle all privacy requirements of HIPAA and GLBA. In this work, we combine the space- and time-efficiency of online monitoring with the generality of offline monitoring: We extend existing work in online monitoring [5] for privacy policy violations with a brute force search fallback based on offline audit for subformulas that are not amenable to caching. Like the work of Basin *et al.* [5], our work uses policies written in metric first-order temporal logic (MFOTL) [16].

Our key technical innovation is what we call the *temporal mode check*, a new static check on formulas to ensure finiteness of quantifier instantiation in our algorithm. Like a standard mode check, the temporal mode check is flow-sensitive: It can propagate variable groundedness information across subformulas. Additionally, the temporal mode check is *time-sensitive*: It conservatively approximates whether the grounding substitution for a variable comes from the future or the past. This allows us to classify all subformulas into those for which we build summary structures during online monitoring (we call such formulas buildable or **B-formulas**) and those for which we do not build summary structures and, hence, use brute force search.

As an example, consider the formula $\Box\exists x, y, z.(\mathsf{p}(x) \wedge \diamondminus\mathsf{q}(x, y) \wedge \diamondminus\mathsf{r}(x, z))$, which means that in all states, there exist x, y, z such that $\mathsf{p}(x)$ holds and in some past states $\mathsf{q}(x, y)$ and $\mathsf{r}(x, z)$ hold. Assume that p and q are finite predicates and that r is infinite, but given a ground value for its first argument, the second argument has finite computable support. One possible efficient strategy for monitoring this formula is to build summary structures for p and q and in each state where an x satisfying p exists, to quickly *lookup* the summary structure for q to find a past state and a y such that $\diamondminus\mathsf{q}(x, y)$ holds, and to *scan* the log brute force to find a past state and z such that $\diamondminus\mathsf{r}(x, z)$ holds. Note that doing so requires marking p and q as **B-formulas**, but r as not a **B-formula** (because z can be computed only after x is known, but x is known from satisfaction of p, which happens in the *future* of r). Unlike the safe-range check or the standard mode check, our new temporal mode check captures this information correctly and our monitoring algorithm, **précis**, implements this strategy. No existing work on online monitoring can handle this formula because r cannot be summarized [5–11]. The work on offline checking can handle this formula [4], it does not build summary structures and is needlessly inefficient on q.

We prove the correctness of **précis** over formulas that pass the temporal mode check and analyze its asymptotic complexity. We also empirically evaluate the performance of **précis** on synthetically generated traces, with respect to privacy policies derived from HIPAA and GLBA. The goal of our experiment is to demonstrate that incrementally maintaining summary structures for **B-formulas** of the policy can improve the performance of policy compliance checking relative to a baseline of pure brute force search. This baseline algorithm is very similar to the offline monitoring algorithm of [4], called **reduce**. In our experiments, we observe marked improvements in running time over **reduce**, e.g., up to 2.5x-6.5x speedup for HIPAA and up to 1.5x speed for GLBA, even with very conservative (unfavorable) assumptions about disk access. Even though these speedups are not universal (online monitoring optimistically constructs summary structures and if those structures are not used later then computation is wasted), they do indicate that temporal mode checking and our monitoring algorithm could have substantial practical benefit for privacy policy compliance.

2 Policy Specification Logic

Our policy specification logic, $\mathcal{GMP}$, is a fragment of MFOTL [16, 17] with restricted universal quantifiers. The syntax of $\mathcal{GMP}$ is shown below.

$$\begin{array}{lcl} (\textbf{Policy formula}) \quad \varphi & ::= & \mathsf{p}(\vec{t}) \mid \top \mid \bot \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \exists\vec{x}.\varphi \mid \forall\vec{x}.(\varphi_1 \to \varphi_2) \\ & & \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 \mid \diamondminus_{\mathbb{I}}\varphi \mid \boxminus_{\mathbb{I}}\varphi \mid \ominus_{\mathbb{I}}\varphi \mid \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 \mid \Diamond_{\mathbb{I}}\varphi \mid \Box_{\mathbb{I}}\varphi \mid \bigcirc_{\mathbb{I}}\varphi \end{array}$$

The letter t denotes terms, which are *constants* or *variables* (x, y, etc.). Roman letters with right arrows on the top like $\vec{t}$ denote sequences or vectors. Policy formulas are denoted by φ, α, and β. Universal quantifiers have a restricted form $\forall\vec{x}.\varphi_1 \to \varphi_2$. A *guard* [18] φ_1 is required as explained further in Section 3.

Policy formulas include both past temporal operators ($\diamondminus$, $\boxminus$, $\mathcal{S}$, $\ominus$) and future temporal operators ($\Diamond$, $\Box$, $\mathcal{U}$, $\bigcirc$). Each temporal operator has an associated time interval $\mathbb{I}$ of the form

$[lo, hi]$, where $lo, hi \in \mathbb{N}$ and $lo \leq hi$. The interval selects a sub-part of the trace in which the immediate subformula is interpreted. For example, $\diamondminus_{[2,6]}\varphi$ means that at some point between 2 and 6 time units in the past, φ holds. For past temporal operators, we allow the higher limit (hi) of $\mathbb{I}$ to be ∞. We omit the interval when it is $[0, \infty]$. Policies must be *future-bounded*: both limits (lo and hi) of intervals associated with future temporal operators must be finite. $\mathcal{GMP}$ is not closed under negation due to the absence of the duals of operators $\mathcal{S}$ and $\mathcal{U}$. However, these operators do not arise in the practical privacy policies we have investigated.

Formulas are interpreted over a timed event trace (or, log) $\mathcal{L}$. Given a possibly-infinite domain of terms $\mathcal{D}$, each element of $\mathcal{L}$—the ith element is denoted $\mathcal{L}_i$— maps each ground atom $\mathsf{p}(\vec{t})$ for $\vec{t} \in \vec{\mathcal{D}}$ to either true or false. Each position $\mathcal{L}_i$ is associated with a time stamp, $\tau_i \in \mathbb{N}$, which is used to interpret intervals in formulas. We use τ to represent the sequence of time stamps, each of which is a natural number. For any arbitrary $i, j \in \mathbb{N}$ with $i > j$, $\tau_i > \tau_j$ (monotonicity). The environment η maps free variables to values in $\mathcal{D}$. Given an execution trace $\mathcal{L}$ and a time stamp-sequence τ, a position $i \in \mathbb{N}$ in the trace, an environment η, and a formula φ, we write $\mathcal{L}, \tau, i, \eta \models \varphi$ to mean that φ is satisfied in the ith position of $\mathcal{L}$ with respect to η and τ. The definition of $\models$ is standard and is presented below. Note that, given an interval $\mathbb{I} = [lo, hi]$ where $lo, hi \in \mathbb{N}$ and $lo \leq hi$, we write $d \in \mathbb{I}$ if it satisfies the following: $lo \leq d \leq hi$.

- $\mathcal{L}, \tau, i, \eta \models \top$ and $\mathcal{L}, \tau, i, \eta \not\models \bot$
- $\mathcal{L}, \tau, i, \eta \models p(\vec{t})$ *iff* $\mathcal{L}_i(p(\eta(\vec{t})))$ is true.
- $\mathcal{L}, \tau, i, \eta \models \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{L}, \tau, i, \eta \models \varphi_1$ and $\mathcal{L}, \tau, i, \eta \models \varphi_2$.
- $\mathcal{L}, \tau, i, \eta \models \varphi_1 \vee \varphi_2$ *iff* $\mathcal{L}, \tau, i, \eta \models \varphi_1$ or $\mathcal{L}, \tau, i, \eta \models \varphi_2$.
- $\mathcal{L}, \tau, i, \eta \models \exists \vec{x}.\varphi$ *iff* there exists $\vec{t}$ such that $\mathcal{L}, \tau, i, \eta[\vec{x} \mapsto \vec{t}] \models \varphi$.
- $\mathcal{L}, \tau, i, \eta \models \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$ *iff* for all $\vec{t}$ if $\mathcal{L}, \tau, i, \eta[\vec{x} \mapsto \vec{t}] \models \varphi_1$ holds then $\mathcal{L}, \tau, i, \eta[\vec{x} \mapsto \vec{t}] \models \varphi_2$ holds.
- $\mathcal{L}, \tau, i, \eta \models \diamondminus_{\mathbb{I}}\varphi$ *iff* there exists $k \leq i$, where $k \in \mathbb{N}$, such that $(\tau_i - \tau_k) \in \mathbb{I}$ and $\mathcal{L}, \tau, k, \eta \models \varphi$.
- $\mathcal{L}, \tau, i, \eta \models \boxminus_{\mathbb{I}}\varphi$ *iff* for all $k \leq i$, where $k \in \mathbb{N}$, such that $(\tau_i - \tau_k) \in \mathbb{I}$, $\mathcal{L}, \tau, k, \eta \models \varphi$ holds.
- $\mathcal{L}, \tau, i, \eta \models \ominus_{\mathbb{I}}\varphi$ *iff* $i > 0$, $\mathcal{L}, \tau, i-1, \eta \models \varphi$, and $\tau_i - \tau_{i-1} \in \mathbb{I}$.
- $\mathcal{L}, \tau, i, \eta \models \varphi_1 \, \mathcal{S}_{\mathbb{I}} \varphi_2$ *iff* there exists $k \leq i$, where $k \in \mathbb{N}$, such that $(\tau_i - \tau_k) \in \mathbb{I}$ and $\mathcal{L}, \tau, k, \eta \models \varphi_2$ and for all j, where $j \in \mathbb{N}$ and $k < j \leq i$, it implies that $\mathcal{L}, \tau, j, \eta \models \varphi_1$ holds.
- $\mathcal{L}, \tau, i, \eta \models \Diamond_{\mathbb{I}}\varphi$ *iff* there exists $k \geq i$, where $k \in \mathbb{N}$, such that $(\tau_k - \tau_i) \in \mathbb{I}$ and $\mathcal{L}, \tau, k, \eta \models \varphi$.
- $\mathcal{L}, \tau, i, \eta \models \Box_{\mathbb{I}}\varphi$ *iff* for all $k \geq i$, where $k \in \mathbb{N}$, such that $(\tau_k - \tau_i) \in \mathbb{I}$, $\mathcal{L}, \tau, k, \eta \models \varphi$ holds.
- $\mathcal{L}, \tau, i, \eta \models \bigcirc_{\mathbb{I}}\varphi$ *iff* $\mathcal{L}, \tau, i+1, \eta \models \varphi$, and $\tau_{i+1} - \tau_i \in \mathbb{I}$.
- $\mathcal{L}, \tau, i, \eta \models \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2$ *iff* there exists $k \geq i$, where $k \in \mathbb{N}$, such that $(\tau_k - \tau_i) \in \mathbb{I}$ and $\mathcal{L}, \tau, k, \eta \models \varphi_2$ and for all j, where $j \in \mathbb{N}$ and $i \leq j < k$, it implies that $\mathcal{L}, \tau, j, \eta \models \varphi_1$ holds.

Example policy. The following $\mathcal{GMP}$ formula represents a privacy rule from clause §6802(a) of the U.S. privacy law GLBA [2]. It states that a financial institution can disclose to a non-affiliated third party any non-public personal information (*e.g.*, name, SSN) if such financial institution provides (within 30 days) or has provided, to the consumer, a notice of the disclosure.

$$\begin{aligned}
&\forall p_1, p_2, q, m, t, u, d.\,(\ \mathsf{send}(p_1^-, p_2^-, m^-) \wedge \mathsf{contains}(m^+, q^-, t^-) \wedge \mathsf{info}(m^+, d^-, u^-) \rightarrow \\
&\quad \mathsf{inrole}(p_1^-, \mathit{institution}^+) \wedge \mathsf{nonAffiliate}(p_2^+, p_1^+) \wedge \mathsf{consumerOf}(q^-, p_1^+) \wedge \mathsf{attrIn}(t, npi) \\
&\qquad \wedge \diamondminus(\exists m_1.\mathsf{send}(p_1^-, q^-, m_1^-) \wedge \mathsf{noticeOfDisclosure}(m_1^+, p_1^+, p_2^+, q^+, t^+)\ ) \vee \\
&\qquad\quad \Diamond_{[0,30]}\exists m_2.\mathsf{send}(p_1^-, q^-, m_2^-) \wedge \mathsf{noticeOfDisclosure}(m_2^+, p_1^+, p_2^+, q^+, t^+)\ )
\end{aligned}$$

3 Temporal Mode Checking

We review mode-checking and provide an overview of our key insight, temporal mode-checking. Then, we define temporal mode-checking for $\mathcal{GMP}$ formally.

3.1 Mode Checking

Consider a predicate $\mathsf{addLessEq}(x, y, a)$, meaning $x+y \leq a$, where x, y, and a range over $\mathbb{N}$. If we are given ground values for x and a, then the number of substitutions for y for which $\mathsf{addLessEq}(x, y, a)$ holds is finite. In this case, we may say that $\mathsf{addLessEq}$'s argument position 1 and 3 are input positions (denoted by '+') and argument position 2 is an output position (denoted by '−'), denoted $\mathsf{addLessEq}(x^+, y^-, a^+)$. Such a specification of inputs and outputs is called a *mode-specification.* The meaning of a mode-specification for a predicate is that if we are given ground values for arguments in the input positions, then the number of substitutions for the variables in the output positions that result in a satisfied relation is finite. For instance, $\mathsf{addLessEq}(x^+, y^+, a^-)$ is not a valid mode-specification. Mode analysis (or mode-checking) lifts input-output specifications on predicates to input-output specification on formulas. It is commonly formalized as a judgment $\chi_{in} \vdash \varphi : \chi_{out}$, which states that given a grounding substitution for variables in χ_{in}, there is at most a finite set of substitutions for variables in χ_{out} that could together satisfy φ. For instance, consider the formula $\varphi \equiv \mathsf{p}(x) \wedge \mathsf{q}(x, y)$. Given the mode-specification $\mathsf{p}(x^-)$ and $\mathsf{q}(x^+, y^-)$ and a left-to-right evaluation order for conjunction, φ passes mode analysis with $\chi_{in} = \{\}$ and $\chi_{out} = \{x, y\}$. Mode analysis guides an algorithm to obtaining satisfying substitutions. In our example, we first obtain substitutions for x that satisfy $\mathsf{p}(x)$. Then, we plug ground values for x in $\mathsf{q}(x, y)$ to get substitutions for y. However, if the mode-specification is $\mathsf{p}(x^+)$ and $\mathsf{q}(x^+, y^-)$, then φ will fail mode analysis unless x is already ground (*i.e.*, $x \in \chi_{in}$).

Mode analysis can be used to identify universally quantified formulas whose truth is finitely checkable. We only need to restrict universal quantifiers to the form $\forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$, and require that $\vec{x}$ be in the output of φ_1 and that φ_2 be well-moded (x may be in its input). To check that $\forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$ is true, we first find the values of $\vec{x}$ that satisfy φ_1. This is a finite set because $\vec{x}$ is in the output of φ_1. We then check that for each of these $\vec{x}$'s, φ_2 is satisfied.

3.2 Overview of Temporal Mode Checking

Consider the policy $\varphi_p \equiv \mathsf{p}(x^-) \wedge \diamondminus\mathsf{q}(x^+, y^-)$ and consider the following obvious but inefficient way to monitor it: We wait for $\mathsf{p}(x)$ to hold for some x, then we look back in the trace to find a position where $\mathsf{q}(x, y)$ holds for some y. This is mode-compliant (we only check q with its input x ground) but requires us to traverse the trace backward whenever $\mathsf{p}(x)$ holds for some x, which can be slow.

Ideally, we would like to incrementally build a summary structure for $\diamondminus \mathsf{q}(x, y)$ containing all the substitutions for x and y for which the formula holds as the monitor processes each new trace event. When we see $\mathsf{p}(x)$, we could quickly look through the summary structure to check whether a relation of the form $\mathsf{q}(x, y)$ for the specific x and any y exists. However, note that building such a structure may be *impossible* here. Why? The mode-specification $\mathsf{q}(x^+, y^-)$ tells us only that we will obtain a finite set of satisfying substitutions when x is already ground. However, in this example, the ground x comes from p, which holds in the *future* of q, so the summary structure may be infinite and, hence, unbuildable. In contrast, if the mode-specification of q is $\mathsf{q}(x^-, y^-)$, then we can build the summary structure because, independent of whether or not x is ground, only a finite number of substitutions can satisfy q. In this example, we would label $\diamondminus \mathsf{q}(x, y)$ *buildable* or a **`B-formula`** when the mode-specification is $\mathsf{q}(x^-, y^-)$ and a non-**`B-formula`** when the mode-specification is $\mathsf{q}(x^+, y^-)$.

With conventional mode analysis, φ_p is well-moded under both mode-specifications of q. Consequently, in order to decide whether φ_p is a **`B-formula`**, we need a refined analysis which takes into account the fact that, with the mode-specification $\mathsf{q}(x^+, y^-)$, information about grounding of x flows *backward* in time from p to q and, hence, $\diamondminus \mathsf{q}(x, y)$ is not a **`B-formula`**. This is precisely what our temporal mode-check accomplishes: It tracks whether an input substitution comes from the past/current state, or from the future. By doing so, it provides enough information to determine which subformulas are **`B-formulas`**.

Formally, our temporal mode-checking has two judgments: $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$ and $\chi_C, \chi_F \vdash \varphi : \chi_O$. The first judgment assumes that substitutions for χ_C are available from the past or at the current time point; any subformula satisfying such a judgment is labeled as a **`B-formula`**. The second judgment assumes that substitutions for χ_C are available from the past or at current time point, but those for χ_F will be available in future. A formula satisfying such a judgment is not a **`B-formula`** but can be handled by brute force search. Our implementation of temporal mode analysis first tries to check a formula by the first judgment, and falls back to the second when it fails. The formal rules for mode analysis (described later) allow for both possibilities but do not prescribe a preference. At the top-level, φ is *well-moded* if $\{\}, \{\} \vdash \varphi : \chi_O$ for some χ_O.

To keep things simple, we do not build summary structures for future formulas such as $\alpha \, \mathcal{U}_{\mathbb{I}} \beta$, and do not allow future formulas in the judgment form $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$ (however, we do build summary structures for nested past-subformulas of future formulas). To check $\alpha \, \mathcal{U}_{\mathbb{I}} \beta$, we wait until the upper limit of $\mathbb{I}$ is exceeded and then search backward. As an optimization, one may build conservative summary structures for future formulas, as in some prior work [5].

3.3 Recognizing `B-formulas`

We list selected rules of temporal mode-checking in Figure 1. The complete list of rules of temporal mode checking can be found in Appendix A. Rule B-Pre, which applies to an atom $\mathsf{p}(t_1, \ldots, t_n)$, checks that all variables in input positions of p are in χ_C. The output χ_O is the set of variables in output positions of p. ($I(\mathsf{p})$ and $O(\mathsf{p})$ are the sets of input and output positions of p, respectively.) The rule for conjunctions $\varphi_1 \wedge \varphi_2$ first checks φ_1 and then checks φ_2, propagating variables in the output of φ_1 to the input of φ_2. These two rules are standard in mode-checking. The new, interesting rule is B-Since for the formula $\varphi_1 \, \mathcal{S}_{\mathbb{I}} \varphi_2$. Since structures for φ_1 and φ_2 could be built

$\boxed{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O}$

$$\dfrac{\forall k \in I(\mathsf{p}).fv(t_k) \subseteq \chi_C \qquad \chi_O = \bigcup\limits_{j \in O(\mathsf{p})} fv(t_j)}{\chi_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O}\ \text{B-PRE}$$

$$\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cup \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \wedge \varphi_2 : \chi_O}\ \text{B-AND}$$

$$\dfrac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_1 \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ \text{B-SINCE}$$

$\boxed{\chi_C, \chi_F \vdash \varphi : \chi_O}$

$$\dfrac{\forall k \in I(\mathsf{p}).fv(t_k) \subseteq (\chi_C \cup \chi_F) \qquad \chi_O = \bigcup\limits_{j \in O(\mathsf{p})} fv(t_j)}{\chi_C, \chi_F \vdash \mathsf{p}(t_1, \ldots, t_n) : \chi_O}\ \text{PRE}$$

$$\dfrac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_1, \chi_C \cup \chi_F \vdash \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ \text{SINCE-1}$$

$$\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}\ \text{UNTIL-1}$$

$$\dfrac{\begin{array}{c}\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \qquad \{\vec{x}\} \subseteq \chi_1 \\ fv(\varphi_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\} \qquad fv(\varphi_2) \subseteq (\chi_C \cup \chi_1 \cup \chi_F) \\ \chi_C, \chi_F \cup \chi_1 \vdash \varphi_2 : \chi_2\end{array}}{\chi_C, \chi_F \vdash \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) : \{\}}\ \text{UNIV-1}$$

Figure 1: Selected rules of temporal mode-checking

at time points earlier than the current time, the premise simply ignores the input χ_C. The first premise of B-SINCE checks φ_2 with an empty input. Based on the semantics of temporal logic, φ_1 needs to be true on the trace after φ_2, so all variables ground by φ_2 (*i.e.*, χ_1) are available as "current" input in φ_1. As an example, $\{\} \vdash_{\mathbf{B}} \top \mathcal{S}\, \mathsf{q}(x^-, y^-) : \{x, y\}$.

3.4 Temporal Mode Checking Judgement

In the mode-checking judgment $\chi_C,\ \chi_F \vdash \varphi : \chi_O$, we separate the set of input variables for which substitutions are available at the current time point or from the past (χ_C) from the set of variables for which substitutions are available from the future (χ_F). The distinction is needed because subderivations of the form $\chi'_C \vdash_{\mathbf{B}} \varphi' : \chi'_O$ should be passed only the former variables as input. Please note that the complete list of rules of temporal mode checking can be found in Appendix A.

Rule PRE for atoms checks that variables in input positions are in the union of χ_C and χ_F. There are four rules for $\varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$, accounting for the buildability/non-buildability of each of the two subformulas. We show only one of these four rules, SINCE-1, which applies when φ_2 is a **`B-formula`** but φ_1 is not. In this case, φ_2 will be evaluated (for creating the summary structure) at time points earlier than $\varphi_1 \mathcal{S}\, \varphi_2$ and, therefore, cannot use variables in χ_C or χ_F as input (see Figure 2). When checking φ_1, variables in the output of φ_2 (called χ_1), χ_C and χ_F are all inputs, but those in χ_C or χ_F come from the future. The entire formula is not a **B-formula** as φ_1 is not.

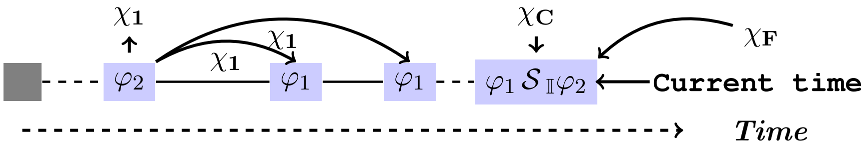


Figure 2: Example: Temporal information in mode checking $\varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$

Similarly, there are four rules for $\varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2$, of which we show only one, UNTIL-1. This rule applies when φ_2 is a **B-formula**, but φ_1 is not. Its first premise checks that φ_2 is a **B-formula** with input χ_C. Our algorithm checks φ_1 only when φ_2 is true, so the outputs χ_1 of φ_2 are available as input for φ_1. In checking φ_1, both χ_1 and χ_F may come from the future.

The first premise of rule UNIV-1 checks that the guard φ_1 is well-moded with some output χ_1. The second premise, $\{\vec{x}\} \subseteq \chi_1$, ensures that the guard φ_1 can be satisfied only for a finite number of substitutions for $\vec{x}$, which is necessary to feasibly check φ_2. The third premise, $fv(\varphi_1) \subseteq (\chi_C \cup \chi_F \cup \{\vec{x}\})$, ensures that no variables other than $\vec{x}$ are additionally grounded by checking φ_1. The fourth premise, $fv(\varphi_2) \subseteq (\chi_C \cup \chi_F \cup \chi_1)$, ensures that all free variables in φ_2 are already grounded by the time φ_2 needs to be checked. The final premise ensures the well-modedness of φ_2. The third and fourth premises are technical conditions, needed for the soundness of our algorithm.

4 Runtime Monitoring Algorithm

Our policy compliance algorithm **précis** takes as input a well-moded $\mathcal{GMP}$ policy φ, monitors the system trace as it grows, builds summary structures for nested **B-formula**s and reports a violation as soon as it is detected.

We write σ to denote a substitution, a finite map from variables to values in the domain $\mathcal{D}$. The identity substitution is denoted $\bullet$ and $\sigma_\perp$ represents an invalid substitution. For instance, the result of joining ($\bowtie$) two substitutions σ_1 and σ_2 that do not agree on the values of shared variables is $\sigma_\perp$. We say that σ' extends σ, written $\sigma' \geq \sigma$, if the domain of σ' is a superset of the domain of σ and they agree on mappings of variables that are in the domain of σ. We summarize relevant algorithmic functions below.

précis(φ) is the top-level function (Algorithm 1).

checkCompliance($\mathcal{L}, i, \tau, \pi, \varphi$) checks whether events in the ith position of the trace $\mathcal{L}$ satisfy φ, given the algorithm's internal state π and the time stamps τ. State π contains up-to-date summary structures for all **B-formulas** of φ.

uSS($\mathcal{L}, i, \tau, \pi, \varphi$) incrementally updates summary structures for **B-formula** φ when log position i is seen. It assumes that the input π is up-to-date w.r.t. earlier log positions and it returns the state with the updated summary structure for φ. (**uSS** is the abbreviation of **updateSummaryStructures**).

sat($\mathcal{L}, i, \tau, \mathsf{p}(\vec{t}), \sigma$) returns the set of all substitutions σ_1 for free variables in $\mathsf{p}(\vec{t})$ that make $\mathsf{p}(\vec{t})\sigma_1$ true in the ith position of $\mathcal{L}$, given σ that grounds variables in the input positions of p. Here, $\sigma_1 \geq \sigma$.

ips$(\mathcal{L}, i, \tau, \pi, \sigma, \varphi)$ generalizes **sat** from atomic predicates to policy formulas. It takes the state π as an input to look up summary structures when **B-formulas** are encountered.

4.1 Top-level monitoring algorithm.

Algorithm 1 (**précis**), the top-level monitoring process, uses two pointers to log entries: *curPtr* points to the last entry in the log $\mathcal{L}$, and *evalPtr* points to the position at which we next check whether φ is satisfied. Naturally, $\mathit{curPtr} \geq \mathit{evalPtr}$. The gap between these two pointers is determined by the intervals occurring in future temporal operators in φ. For example, with the policy $\Diamond_{[lo,hi]}\beta$, β can be evaluated at log position i only after a position $j \geq i$ with $\tau_j - \tau_i \geq hi$ has been observed. We define a simple function $\Delta(\varphi)$ just below that computes a coarse but finite upper bound on the maximum time the monitor needs to wait before φ can be evaluated. We want to emphasize that future boundedness of $\mathcal{GMP}$ policies and monotonicity requirement on τ ensure that $\Delta(\varphi)$ is finite and bounded.

$$\Delta(\varphi) = \begin{cases} 0 & \text{if } \varphi \equiv \top \mid \bot \mid \mathsf{p}(t_1, \ldots, p_n) \\ max(\Delta(\varphi_1), \Delta(\varphi_2)) & \text{if } \varphi \equiv \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) \mid \varphi_1 \,\mathcal{S}_{[c,d]} \varphi_2 \\ \Delta(\varphi) & \text{if } \varphi \equiv \exists \vec{x}.\varphi \mid \diamondminus_{[c,d]}\varphi \mid \boxminus_{[c,d]}\varphi \mid \ominus_{[c,d]}\varphi \\ d + max(\Delta(\varphi_1), \Delta(\varphi_2)) & \text{if } \varphi \equiv \varphi_1 \,\mathcal{U}_{[c,d]} \varphi_2 \\ d + \Delta(\varphi) & \text{if } \varphi \equiv \Diamond_{[c,d]}\varphi \mid \Box_{[c,d]}\varphi \mid \bigcirc_{[c,d]}\varphi \end{cases}$$

Figure 3: Definition of $\Delta(\varphi)$

Algorithm 1 The **précis** algorithm

```
Require: A $\mathcal{GMP}$ policy $\varphi$
 1: $\pi \leftarrow \emptyset$;   $curPtr \leftarrow 0$;   $evalPtr \leftarrow 0$;   $\mathcal{L} \leftarrow \emptyset$;   $\tau \leftarrow \emptyset$;
 2: Mode-check $\varphi$.  Label all B-formulas of $\varphi$.
 3: while ($true$) do
 4:   Wait until new events are available
 5:   Extend $\mathcal{L}$ and $\tau$ with new entries
 6:   for all (B-formulas $\varphi_s$ of $\varphi$ in ascending formula size) do
 7:     $\pi \leftarrow$ uSS($\mathcal{L}, curPtr, \tau, \pi, \varphi_s$)    //update summary structures
 8:   while ($evalPtr \leq curPtr$) do
 9:     if ($\tau_{curPtr} - \tau_{evalPtr} \geq \Delta(\varphi)$) then
10:       $tVal \leftarrow$ checkCompliance($\mathcal{L}, evalPtr, \tau, \pi, \varphi$)
11:       if $tVal = false$ then
12:         Report violation on $\mathcal{L}$ position $evalPtr$
13:       $evalPtr \leftarrow evalPtr + 1$
14:     else
15:       break
16:   $curPtr \leftarrow curPtr + 1$
```

The algorithm **précis** first initializes relevant data structures and labels **B-formulas** using mode analysis (lines 1-2). The main body of the **précis** is a trace-event triggered loop. In

$$\texttt{\textbf{checkCompliance}}(\mathcal{L}, i, \tau, \pi, \varphi) = \begin{cases} true & \text{if } \texttt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \bullet, \varphi) \neq \{\} \\ false & \text{otherwise} \end{cases}$$

Figure 4: Definition of the **checkCompliance** function.

each iteration of the loop, **précis**: (1) updates the summary structures in π based on the newly available log entries (lines 6-7), and (2) evaluates the policy at positions where it can be fully evaluated, i.e., where the difference between the entry's time point and the current time point ($curPtr$) exceeds the maximum delay $\Delta(\varphi)$. Step (1) uses the function **uSS** and step (2) uses the function **checkCompliance** (see Figure 4). **checkCompliance** is a wrapper for **ips** that calls **ips** with $\bullet$ as the input substitution. If **ips** returns an empty set of satisfying substitutions, **checkCompliance** returns false, signaling a violation at the current time point, else it returns true.

4.2 Finding substitutions for policy formulas.

The recursive function **ips** returns the set of substitutions that satisfy a formula at a given log position, given a substitution for the formula's input variables. Selected clauses of the definition of **ips** are shown in Figure 5. All the clauses of the definition of **ips** can be found in Appendix B. When the formula is an atom, **ips** invokes **sat**, an abstract wrapper around specific implementations of predicates. When the policy is a universally quantified formula, **ips** is called on the guard φ_1 to find the guard's satisfying substitutions Σ_1. Then, **ips** is called to check that φ_2 is true for all substitutions in Σ_1. If the latter fails, **ips** returns the empty set of substitutions to signal a violation, else it returns $\{\sigma_{in}\}$.

When a **B-formula** $\alpha\,\mathcal{S}_{\mathbb{I}}\beta$ is encountered, all its satisfying substitutions have already been computed and stored in π. Therefore, **ips** simply finds these substitutions in π (expression $\pi.\mathcal{A}(\alpha\,\mathcal{S}_{\mathbb{I}}\beta)(i).\mathbb{R}$), and discards those that are inconsistent with σ_{in} by performing a join ($\bowtie$). For the non-**B-formula** $\alpha\,\mathcal{S}_{\mathbb{I}}\beta$, **ips** calls itself recursively on the sub-formulas α and β, and computes the substitutions brute force.

4.3 Incrementally updating summary structures.

We explain how we update summary structures for formulas of the form $\varphi_1\,\mathcal{S}_{\mathbb{I}}\varphi_2$ here. Updates for $\ominus_{\mathbb{I}}\varphi$, $\boxminus_{\mathbb{I}}\varphi$, and $\diamondminus_{\mathbb{I}}\varphi$ are similar and can be found in Appendix C.

For each **B-formula** of the form $\alpha\,\mathcal{S}_{[lo,hi]}\beta$, we build three structures: $\mathbb{S}_\beta$, $\mathbb{S}_\alpha$, and $\mathbb{R}$. The structure $\mathbb{S}_\beta$ contains a set of pairs of form $\langle\sigma, k\rangle$ in which σ represents a substitution and $k \in \mathbb{N}$ is a position in $\mathcal{L}$. Each pair of form $\langle\sigma, k\rangle \in \mathbb{S}_\beta$ represents that for all $\sigma' \geq \sigma$, the formula $\beta\sigma'$ is true at position k of $\mathcal{L}$. The structure $\mathbb{S}_\alpha$ contains a set of pairs of form $\langle\sigma, k\rangle$, each of which represents that for all $\sigma' \geq \sigma$ the formula $\alpha\sigma'$ has been true from position k until the current position in $\mathcal{L}$. The structure $\mathbb{R}$ contains a set of substitutions, which make $(\alpha\,\mathcal{S}_{[lo,hi]}\beta)$ true in the current position of $\mathcal{L}$. We use $\mathbb{R}^i$ (similarly for other structures too) to represent the structure $\mathbb{R}$ at position i of $\mathcal{L}$. We also assume $\mathbb{S}_\beta^{(-1)}$, $\mathbb{S}_\alpha^{(-1)}$, and $\mathbb{R}^{(-1)}$ to be empty (the same applies for other structures too). We show here how the structures $\mathbb{S}_\beta$ and $\mathbb{R}$ are updated. We defer the description of update of $\mathbb{S}_\alpha$ to Appendix C.

To update the structure $\mathbb{S}_\beta$, we first calculate the set Σ_β of substitutions that make β true at i

$$\mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \mathsf{p}(\vec{t})) \quad = \quad \mathtt{\textbf{sat}}(\mathcal{L}, i, \tau, \mathsf{p}(\vec{t}), \sigma_{\text{in}})$$

$$\begin{array}{l} \mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \\ \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)) \end{array} \quad = \quad \begin{array}{ll} \text{let} & \Sigma_1 \leftarrow \mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1) \\ \text{return} & \begin{cases} \emptyset & \text{if } \exists \sigma_c \in \Sigma_1.(\mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \sigma_c, \varphi_2) = \emptyset) \\ \{\sigma_{\text{in}}\} & \text{otherwise} \end{cases} \end{array}$$

$$\mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \alpha \, \mathcal{S}_{\mathbb{I}} \beta) \quad = \quad \begin{cases} \textbf{If } \alpha \, \mathcal{S}_{\mathbb{I}} \beta \text{ is a } \mathtt{\textbf{B-formula}} \textbf{ then} \\ \qquad \textbf{return } \sigma_{\text{in}} \bowtie \pi.\mathcal{A}(\alpha \, \mathcal{S}_{\mathbb{I}} \beta)(i).\mathbb{R} \\ \textbf{Else} \\ \text{let} \quad S_\beta \leftarrow \{\langle \sigma, k \rangle | k = \max l.((0 \leq l \leq i) \wedge ((\tau_i - \tau_l) \in \mathbb{I}) \\ \qquad\qquad \wedge \sigma \in \mathtt{\textbf{ips}}(\mathcal{L}, l, \tau, \pi, \sigma_{\text{in}}, \beta))\} \\ \qquad S_{R_1} \leftarrow \{\sigma | \langle \sigma, i \rangle \in S_\beta \wedge 0 \in \mathbb{I}\} \\ \qquad S_{R_2} \leftarrow \{\bowtie \sigma_l^\alpha \neq \sigma_\perp | \exists \langle \sigma_\beta, k \rangle \in S_\beta . k < i \wedge \\ \qquad\qquad\qquad \forall l.(k < l \leq i \rightarrow \sigma_l^\alpha \in \mathtt{\textbf{ips}}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \varphi_1))\} \\ \qquad \textbf{return } S_{R_1} \cup S_{R_2} \end{cases}$$

Figure 5: Definition of the **`ips`** function, selected clauses

by calling **`ips`**. Pairing all these substitutions with the position i yields S^β_{new}. Next, we compute the set S^β_{remove} of all old $\langle \sigma, k \rangle$ pairs that do not satisfy the interval constraint $[lo, hi]$ (i.e., for which $\tau_i - \tau_k > hi$). The updated structure $\mathbb{S}^i_\beta$ is then obtained by taking a union of S^β_{new} and the old structure $\mathbb{S}^{(i-1)}_\beta$, and removing all the pairs in the set S^β_{remove}.

$$\begin{array}{lll|lll} \Sigma_\beta & \leftarrow & \mathtt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \bullet, \beta) & S^\beta_{\text{remove}} & \leftarrow & \{\langle \sigma, k \rangle \mid \langle \sigma, k \rangle \in \mathbb{S}^{(i-1)}_\beta \wedge (\tau_i - \tau_k) > hi\} \\ S^\beta_{\text{new}} & \leftarrow & \{\langle \sigma, i \rangle \mid \sigma \in \Sigma_\beta\} & \mathbb{S}^i_\beta & \leftarrow & (\mathbb{S}^{(i-1)}_\beta \cup S^\beta_{\text{new}}) \setminus S^\beta_{\text{remove}} \end{array}$$

To compute the summary structure $\mathbb{R}$ for $\alpha \, \mathcal{S}_{\mathbb{I}} \beta$ at i, we first compute the set S_{R_1} of all substitutions for which the formula β is true in the ith position and the interval constraint is respected by the position i. Then we compute S_{R_2} as the join $\sigma \bowtie \sigma_1$ of substitutions σ for which β was satisfied at some prior position k, and substitutions σ_1 for which α is true from position $k+1$ to i. The updated structure $\mathbb{R}^i$ is the union of S_{R_1} and S_{R_2}.

$$\begin{array}{lll} S_{R_1} & \leftarrow & \{\sigma \mid \langle \sigma, i \rangle \in \mathbb{S}^i_\beta \wedge 0 \in [lo, hi]\} \\ S_{R_2} & \leftarrow & \{\sigma \bowtie \sigma_1 \mid \exists k, j. \langle \sigma, k \rangle \in \mathbb{S}^i_\beta \wedge (k \neq i) \wedge (\tau_i - \tau_k \in [lo, hi]) \wedge \langle \sigma_1, j \rangle \in \mathbb{S}^i_\alpha \wedge \\ & & \qquad (j \leq (k+1)) \wedge \sigma \bowtie \sigma_1 \neq \sigma_\perp\} \\ \mathbb{R}^i & \leftarrow & S_{R_1} \cup S_{R_2} \end{array}$$

4.4 Optimizations

When all temporal sub-formulas of φ are **`B-formulas`**, *curPtr* and *evalPtr* proceed in synchronization and only the summary structure for position *curPtr* needs to be maintained. When φ contains future temporal formulas but all past temporal sub-formulas of φ are **`B-formulas`**, then we need to maintain only the summary structures for positions in [*evalPtr*, *curPtr*], but the rest of the log can be discarded immediately. When φ contains at least one past temporal subformula that is not a **`B-formula`** we need to store the slice of the trace that contains all predicates in that non-**`B-formula`**.

The following theorem states that on well-moded policies, **`précis`** terminates and is correct.

The theorem requires that the internal state π be *strongly consistent* at $curPtr$ with respect to the log $\mathcal{L}$, time stamp sequence τ, and policy φ. Strong consistency means that the state π contains sound and complete substitutions for all **B-formulas** of φ for all trace positions in $[0, curPtr]$ (see Appendix D.3).

Theorem 1 (Correctness of **précis**). *For all $\mathcal{GMP}$ policies φ, for all $evalPtr$, $curPtr \in \mathbb{N}$, for all traces $\mathcal{L}$, for all time stamp sequences τ, for all internal states π, for all empty environments η_0 such that (1) π is strongly consistent at $curPtr$ with respect to $\mathcal{L}$, τ, and φ, (2) $curPtr \geq evalPtr$ and $\tau_{curPtr} - \tau_{evalPtr} \geq \Delta(\varphi)$, and (3) $\{\}, \{\} \vdash \varphi : \chi_O$ where $\chi_O \subseteq fv(\varphi)$, it is the case that* ***checkCompliance***$(\mathcal{L}, evalPtr, \tau, \pi, \varphi)$ *terminates and if* ***checkCompliance***$(\mathcal{L}, evalPtr, \tau, \pi, \varphi) = tVal$*, then* $(tVal = true) \leftrightarrow \exists\sigma.(\mathcal{L}, \tau, evalPtr, \eta_0 \models \varphi\sigma)$.

Proof. By induction on the policy formula φ (see Appendix D). □

***Complexity of* précis.** The runtime complexity of one iteration of **précis** for a given policy φ is $|\varphi| \times$ (complexity of the **uSS** function) + (complexity of **ips** function), where $|\varphi|$ is the policy size. We first analyze the runtime complexity of **ips**. Suppose the maximum number of substitutions returned by a single call to **sat** (for any position in the trace) is $\mathbb{F}$ and the maximum time required by **sat** to produce one substitution is $\mathbb{A}$. The worst case runtime of **ips** occurs when all subformulas of φ are non-**B-formulas** of the form $\varphi_1 \mathcal{S} \varphi_2$ and in that case the complexity is $\mathcal{O}((\mathbb{A} \times \mathbb{F} \times \mathbb{L})^{\mathcal{O}(|\varphi|)})$ where $\mathbb{L}$ denotes the length of the trace. **uSS** is invoked only for **B-formulas**. From the definition of mode-checking, all sub-formulas of a **B-formula** are also **B-formulas**. This property of **B-formulas** ensures that when **uSS** calls **ips**, the worst case behavior of **ips** is not encountered. The overall complexity of **uSS** is $\mathcal{O}(|\varphi| \times (\mathbb{A} \times \mathbb{F})^{\mathcal{O}(|\varphi|)})$. Thus, the runtime complexity of each iteration of the **précis** function is $\mathcal{O}((\mathbb{A} \times \mathbb{F} \times \mathbb{L})^{\mathcal{O}(|\varphi|)})$.

5 Implementation and Evaluation

This section reports an experimental evaluation of the **précis** algorithm. All measurements were made on a 2.67GHz Intel Xeon CPU X5650 running Debian GNU/Linux 7 (Linux kernel 3.2.48.1.amd64-smp) on 48GB RAM, of which at most 2.2GB is used in our experiments. We store traces in a SQLite database. Each n-ary predicate is represented by a $n+1$ column table whose first n columns store arguments that make the predicate true on the trace and the last column stores the trace position where the predicate is true. We index each table by the columns corresponding to input positions of the predicate. We experiment with randomly generated synthetic traces. Given a $\mathcal{GMP}$ policy and a target trace length, at each trace point, our synthetic trace generator randomly decides whether to generate a policy-compliant action or a policy violating action. For a compliant action, it recursively traverses the syntax of the policy and creates trace actions to satisfy the policy. Disjunctive choices are resolved randomly. Non-compliant actions are handled dually. The source code and traces used in the experiments are available from the authors' homepages.

Our goal is to demonstrate that incrementally maintaining summary structures for **B-formulas** can improve the performance of policy compliance checking. Our baseline for comparison is a variant of **précis** that does not use any summary structures and, hence, checks temporal operators by brute force scanning. This baseline algorithm is very similar to the **reduce** algorithm of prior work [4] and, indeed, in the sequel we refer to our baseline as **reduce**. For the experimental results reported here, we deliberately hold traces in an in-memory SQLite database. This choice

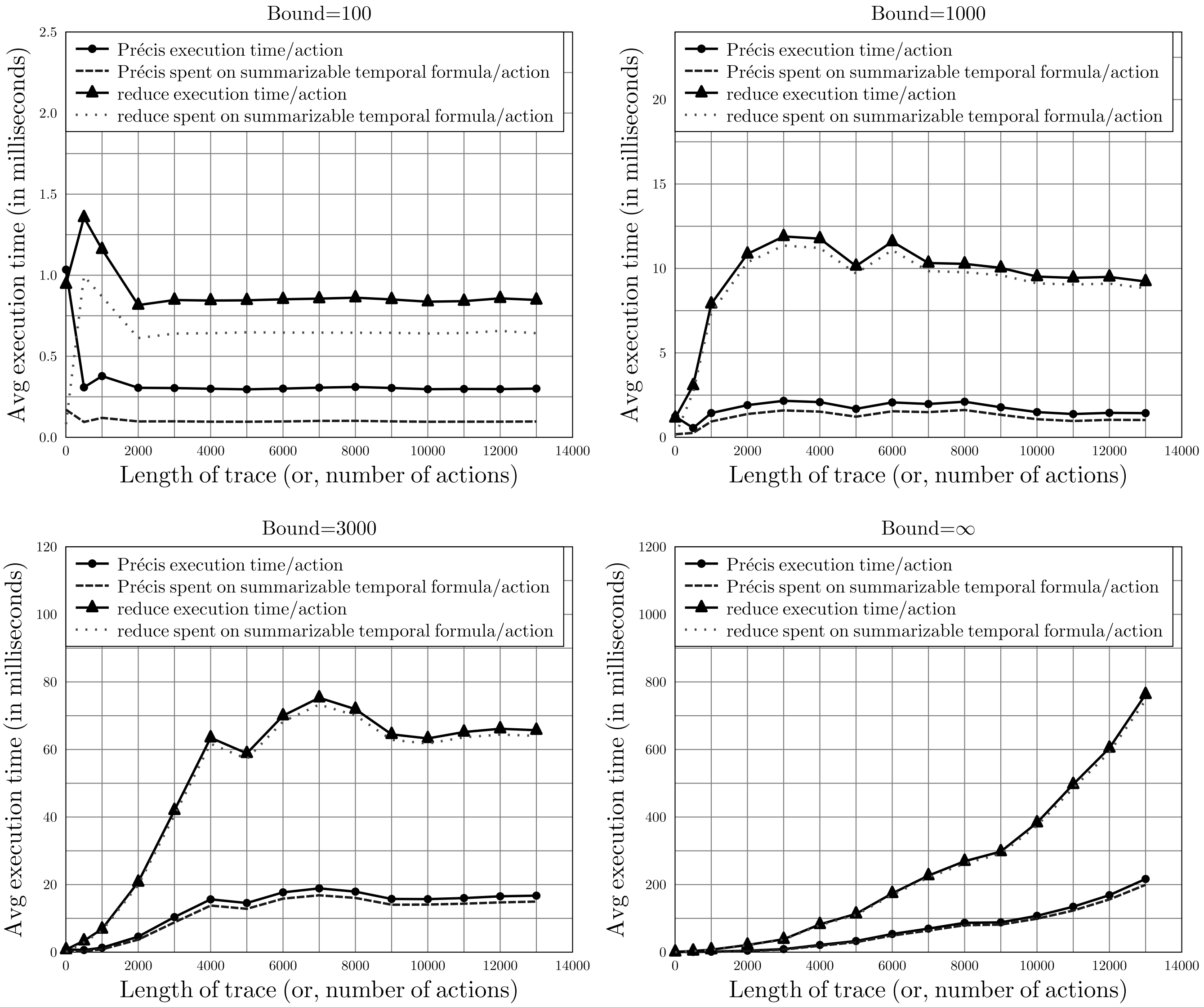


Figure 6: Experimental timing results (HIPAA) with memory-backed database

is conservative; using a disk-backed database improves **précis**' performance relative to **reduce** because **reduce** accesses the database more intensively (Appendix F contains comparative evaluation using a disk-backed database and confirms this claim). Another goal of our experiment is to identify how **précis** scales when larger summary structures must be maintained. Accordingly, we vary the upper bound hi in intervals $[lo, hi]$ in past temporal operators.

We experiment with two privacy policies that contain selected clauses of HIPAA and GLBA, respectively. As **précis** and **reduce** check compliance of non-**B-formulas** similarly, to demonstrate the utility of building summary structures, we ensure that the policies contain **B-formulas**

Algorithms	Incomplete states allowed?	Mode of operation	Summary structures (past formulas)	Summary structures (future formulas)
précis	**no**	**online**	**yes**	**no**
reduce [4]	yes	offline	no	no
Chomicki [8,9] Krukow *et al.* [10]	no	online	yes	no
Bauer *et al.* [11]	yes	online	yes	no
Basin *et al.* [5,7]	no	online	yes	yes
Basin *et al.* [6]	yes	online	yes	yes
Bauer *et al.* [19]	no	online	(automata)*	(automata)*

Table 1: Comparison of design choices in **précis** and prior work using first-order temporal logic for privacy compliance. *Automata-based approaches have no explicit notion of summary structures.

(in our HIPAA policy, 7 out of 8 past temporal formulas are **B-formulas**; for GLBA the number is 4 out of 9). Appendix E lists the policies we used. Figure 6 show our evaluation times for the HIPAA privacy policy for the following upper bounds on the past temporal operators: 100, 1000, 3000, and ∞. Points along the x-axis are the size of the trace and also the number of privacy-critical events checked. The y-axis represents the *average* monitoring time per event. We plot four curves for each bound: (1) The time taken by **précis**, (2) The time taken by **reduce**, (3) The time spent by **précis** in building and accessing summary structures for **B-formulas**, and (4) The time spent by **reduce** in evaluating **B-formulas**. For all trace positions $i \in \mathbb{N}$, $\tau_{i+1} - \tau_i = 1$.

The difference between (1) and (3), and (2) and (4) is similar at all trace lengths because it is the time spent on non-buildable parts of the policy, which is similar in **précis** and **reduce**. For the policy considered here, **reduce** spends most time on **B-formulas**, so construction of summary structures improves performance. For trace lengths greater than the bound, the curves flatten out, as expected. As the bound increases, the average execution time for **reduce** increases as the algorithm has to look back further on the trace, and so does the relative advantage of **précis**. Overall, **précis** achieves a speedup up of 2.5x-6.5x over **reduce** after the curves flatten out in the HIPAA policy. The results for GLBA, not shown here but presented in Appendix F are similar, with speedups of 1.25x to 1.5x. The technical report also describes the amount of memory needed to store summary structures in **précis**. Briefly, this number grows proportional to the minimum of trace length and policy bound. The maximum we observe (for trace length 13000 and bound ∞) is 2.2 GB, which is very reasonable. This can be further improved by compression.

6 Related Work

Runtime monitoring of *propositional* linear temporal logic (pLTL) formulas [20], regular expressions, finite automata, and other equivalent variants has been studied in literature extensively [21–47]. However, pLTL and its variants are not sufficient to capture the privacy requirements of legislation like HIPAA and GLBA. To address this limitation, many logics and languages have been proposed for specifying privacy policies. Some examples are P3P [48, 49], EPAL [50, 51], Privacy APIs [52], LPU [53, 54], past-only fragment of first-order temporal logic (FOTL) [10, 11], predLTL [55], pLogic [56], PrivacyLFP [12], MFOTL [5–7], the guarded fragment of first-order logic with explicit time [4], and P-RBAC [57]. Our policy language, $\mathcal{GMP}$, is more expressive than many existing policy languages such as LPU [53, 54], P3P [48, 49], EPAL [50, 51], and P-RBAC [57].

In Table 1, we summarize design choices in **précis** and other existing work on privacy policy compliance checking using first-order temporal logics. The column "Incomplete states allowed?"

indicates whether the work can handle some form of incompleteness in observation about states. Our own prior work [4] presents the algorithm **reduce** that checks compliance of a mode-checked fragment of FOL policies with respect to potentially incomplete logs. This paper makes the mode check time-aware and adds summary structures to **reduce**, but we assume that our event traces have complete information in all observed states.

Bauer *et al.* [11] present a compliance-checking algorithm for the (non-metric) past fragment of FOTL. $\mathcal{GMP}$ can handle both past and future (metric) temporal operators. However, Bauer *et al.* allow counting operators, arbitrary computable functions, and partial observability of events, which we do not allow. They allow a somewhat simplified guarded universal quantification where the guard is a single predicate. In $\mathcal{GMP}$, we allow the guard of the universal quantification to be a complex $\mathcal{GMP}$ formula. For instance, the following formula cannot be expressed in the language proposed by Bauer *et al.* but $\mathcal{GMP}$ mode checks it: $\forall x, y. (\mathsf{q}(x^+, y^+) \, \mathcal{S} \, \mathsf{p}(x^-, y^-)) \rightarrow \mathsf{r}(x^+, y^+)$. Moreover, Bauer *et al.* only consider closed formulas and also assume that each predicate argument position is output. We do not insist on these restrictions. In further development, Bauer *et al.* [19], propose an automata-based, incomplete monitoring algorithm for a fragment of FOTL called $\mathrm{LTL}^{\mathrm{FO}}$. They consider non-safety policies (unbounded future operators), which we do not consider.

Basin *et al.* [5] present a runtime monitoring algorithm for a fragment of MFOTL. Our summary structures are directly inspired by this work and the work of Chomicki [8, 9]. We improve expressiveness through the possibility of brute force search similar to [4], when subformulas are not amenable to summarization. Basin *et al.* build summary structures for future operators, which we do not (such structures can be added to our monitoring algorithm). In subsequent work, Basin *et al.* [6] extend their runtime monitoring algorithm to handle incomplete logs and inconsistent logs using a three-valued logic, which we do not consider. In more recent work, Basin *et al.* [7] extend the monitoring algorithm to handle aggregation operators and function symbols, which $\mathcal{GMP}$ does not include. These extensions are orthogonal to our work.

Our temporal mode check directly extends mode checking from [4] by adding time-sensitivity, although the setting is different— [4] is based on first-order logic with an explicit theory of linear time whereas we work with MFOTL. The added time-sensitivity allows us to classify subformulas into those that can be summarized and those that must be brute forced. Some prior work, e.g. [5–11], is based on the safe-range check instead of the mode check. The safe-range check is less expressive than a mode check. For example, the safe-range check does not accept the formula $\mathsf{q}(x^+, y^+, z^-) \, \mathcal{S} \, \mathsf{p}(x^-, y^-)$, but our temporal mode check does (however, the safe-range check will accept the formula $\mathsf{q}(x^-, y^-, z^-) \, \mathcal{S} \, \mathsf{p}(x^-, y^-)$). More recent work [7] uses a static check intermediate in expressiveness between the safe-range check and a full-blown mode check.

7 Conclusion

We have presented a privacy policy compliance-checking algorithm for a fragment of MFOTL. The fragment is characterized by a novel temporal mode-check, which, like a conventional mode-check, ensures that only finitely many instantiations of quantifiers are tested but is, additionally, time-aware and can determine which subformulas of the policy are amenable to construction of summary structures. Using information from the temporal mode-check, our algorithm **précis** performs best-effort runtime monitoring, falling back to brute force search when summary structures cannot be constructed. Empirical evaluation shows that summary structures improve performance significantly, compared to a baseline without them.

Acknowledgement

This work was partially supported by the AFOSR MURI on “Science of Cybersecurity”, the National Science Foundation (NSF) grants CNS 1064688, CNS 0964710, and CCF 0424422, and the HHS/ONC grant HHS90TR0003/01. The authors thank anonymous reviewers, Sagar Chaki, Andreas Gampe, and Murillo Pontual for their helpful comments and suggestions.

References

[1] Health Resources and Services Administration: Health insurance portability and accountability act (1996) Public Law 104-191.

[2] Senate Banking Committee: Gramm-Leach-Bliley Act (1999) Public Law 106-102.

[3] Roberts, P.: HIPAA Bares Its Teeth: $4.3m Fine For Privacy Violation Available at https://threatpost.com/en_us/blogs/hipaa-bares-its-teeth-43m-fine-privacy-violation-022311.

[4] Garg, D., Jia, L., Datta, A.: Policy auditing over incomplete logs: Theory, implementation and applications. In: Proceedings of the 18th ACM Conference on Computer and Communications Security. CCS ’11, New York, NY, USA, ACM (2011) 151–162

[5] Basin, D., Klaedtke, F., Müller, S.: Monitoring security policies with metric first-order temporal logic. In: Proceedings of the 15th ACM Symposium on Access Control Models and Technologies. SACMAT ’10, New York, NY, USA, ACM (2010) 23–34

[6] Basin, D., Klaedtke, F., Marinovic, S., Zălinescu, E.: Monitoring compliance policies over incomplete and disagreeing logs. In Qadeer, S., Tasiran, S., eds.: Runtime Verification. Volume 7687 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 151–167

[7] Basin, D., Klaedtke, F., Marinovic, S., Zălinescu, E.: Monitoring of temporal first-order properties with aggregations. In Legay, A., Bensalem, S., eds.: Runtime Verification. Volume 8174 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 40–58

[8] Chomicki, J.: Efficient checking of temporal integrity constraints using bounded history encoding. ACM Trans. Database Syst. **20**(2) (June 1995) 149–186

[9] Chomicki, J., Niwiński, D.: On the feasibility of checking temporal integrity constraints. In: Proceedings of the Twelfth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. PODS ’93, New York, NY, USA, ACM (1993) 202–213

[10] Krukow, K., Nielsen, M., Sassone, V.: A logical framework for history-based access control and reputation systems. J. Comput. Secur. **16**(1) (January 2008) 63–101

[11] Bauer, A., Gore, R., Tiu, A.: A first-order policy language for history-based transaction monitoring. In Leucker, M., Morgan, C., eds.: Proceedings of the 6th International Colloquium on Theoretical Aspects of Computing (ICTAC). Volume 5684 of Lecture Notes in Computer Science., Berlin, Heidelberg, Springer-Verlag (August 2009) 96–111

[12] DeYoung, H., Garg, D., Jia, L., Kaynar, D., Datta, A.: Experiences in the logical specification of the hipaa and glba privacy laws. In: Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society. WPES '10, New York, NY, USA, ACM (2010) 73–82

[13] Apt, K., Marchiori, E.: Reasoning about prolog programs: From modes through types to assertions. Formal Aspects of Computing **6**(1) (1994) 743–765

[14] Dembinski, P., Maluszynski, J.: And-parallelism with intelligent backtracking for annotated logic programs. In: Proceedings of the 1985 Symposium on Logic Programming, Boston, Massachusetts, USA, July 15-18, 1985, IEEE-CS (1985) 29–38

[15] Mellish, C.S.: The automatic generation of mode declarations for Prolog programs. Department of Artificial Intelligence, University of Edinburgh (1981)

[16] Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Systems **2**(4) (1990) 255–299

[17] Alur, R., Henzinger, T.: Logics and models of real time: A survey. In Bakker, J., Huizing, C., Roever, W., Rozenberg, G., eds.: Real-Time: Theory in Practice. Volume 600 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (1992) 74–106

[18] Andréka, H., Németi, I., van Benthem, J.: Modal languages and bounded fragments of predicate logic. Journal of Philosophical Logic **27**(3) (1998) 217–274

[19] Bauer, A., Küster, J.C., Vegliach, G.: From propositional to first-order monitoring. In Legay, A., Bensalem, S., eds.: Runtime Verification. Volume 8174 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 59–75

[20] Pnueli, A.: The temporal logic of programs. In: Proceedings of the 18th Annual Symposium on Foundations of Computer Science. SFCS '77, Washington, DC, USA, IEEE Computer Society (1977) 46–57

[21] Roşu, G.: On Safety Properties and Their Monitoring. Technical Report UIUCDCS-R-2007-2850, Department of Computer Science, University of Illinois at Urbana-Champaign (2007)

[22] Büchi, J.R.: On a Decision Method in Restricted Second-Order Arithmetic. In: International Congress on Logic, Methodology, and Philosophy of Science, Stanford University Press (1962) 1–11

[23] Hussein, S., Meredith, P.O., Roşu, G.: Security-policy monitoring and enforcement with JavaMOP. In: ACM SIGPLAN Seventh Workshop on Programming Languages and Analysis for Security (PLAS'12). (2012) 3:1–3:11

[24] Meredith, P., Roşu, G.: Runtime verification with the RV system. In: First International Conference on Runtime Verification (RV'10). Volume 6418 of Lecture Notes in Computer Science., Springer (2010) 136–152

[25] Meredith, P., Roşu, G.: Efficient parametric runtime verification with deterministic string rewriting. In: Proceedings of 28th IEEE/ACM International Conference. Automated Software Engineering (ASE'13), IEEE/ACM (May 2013) NA

[26] Pellizzoni, R., Meredith, P., Caccamo, M., Roşu, G.: Hardware runtime monitoring for dependable cots-based real-time embedded systems. In: Proceedings of the 29th IEEE Real-Time System Symposium (RTSS'08). (2008) 481–491

[27] Meredith, P., Jin, D., Chen, F., Roşu, G.: Efficient monitoring of parametric context-free patterns. In: Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering(ASE '08), IEEE/ACM (2008) 148–157

[28] Roşu, G., Havelund, K.: Synthesizing dynamic programming algorithms from linear temporal logic formulae. Technical report, Research Institute for Advanced Computer Science (2001)

[29] Roşu, G., Havelund, K.: Rewriting-based techniques for runtime verification. Automated Software Engineering **12**(2) (2005) 151–197

[30] Havelund, K., Roşu, G.: Efficient monitoring of safety properties. Int. J. Softw. Tools Technol. Transf. **6**(2) (August 2004) 158–173

[31] Roşu, G., Chen, F., Ball, T.: Synthesizing monitors for safety properties – this time with calls and returns –. In: Workshop on Runtime Verification (RV'08). Volume 5289 of Lecture Notes in Computer Science., Springer (2008) 51–68

[32] Leucker, M., Schallhart, C.: A brief account of runtime verification. The Journal of Logic and Algebraic Programming **78**(5) (2009) 293 – 303 The 1st Workshop on Formal Languages and Analysis of Contract-Oriented Software (FLACOS'07).

[33] Roşu, G., Bensalem, S.: Allen Linear (Interval) Temporal Logic –Translation to LTL and Monitor Synthesis. In: Proceedings of 18th International Conference on Computer Aided Verification (CAV'06). Volume 4144 of Lecture Notes in Computer Science., Springer (2006) 263–277

[34] d'Amorim, M., Roşu, G.: Efficient monitoring of ω-languages. In: Proceedings of 17th International Conference on Computer-aided Verification (CAV'05). Volume 3576 of Lecture Notes in Computer Science., Springer (2005) 364 – 378

[35] Basin, D., Jugé, V., Klaedtke, F., Zălinescu, E.: Enforceable security policies revisited. ACM Trans. Inf. Syst. Secur. **16**(1) (June 2013) 3:1–3:26

[36] Bauer, L., Ligatti, J., Walker, D.: More enforceable security policies. In Cervesato, I., ed.: Foundations of Computer Security: proceedings of the FLoC'02 workshop on Foundations of Computer Security, Copenhagen, Denmark, DIKU Technical Report (25–26 July 2002) 95–104

[37] Schneider, F.B.: Enforceable security policies. ACM Trans. Inf. Syst. Secur. **3**(1) (February 2000) 30–50

[38] Giannakopoulou, D., Havelund, K.: Automata-based verification of temporal properties on running programs. In: Automated Software Engineering, 2001. (ASE 2001). Proceedings. 16th Annual International Conference on. (Nov 2001) 412–416

[39] Martinell, F., Matteucci, I.: Through modeling to synthesis of security automata. Electron. Notes Theor. Comput. Sci. **179** (2007) 31–46

[40] Huisman, M., Tamalet, A.: A formal connection between security automata and jml annotations. In: FASE '09: Proceedings of the 12th International Conference on Fundamental Approaches to Software Engineering, Berlin, Heidelberg, Springer-Verlag (2009) 340–354

[41] Ligatti, J., Bauer, L., Walker, D.: Run-time enforcement of nonsafety policies. ACM Trans. Inf. Syst. Secur. **12**(3) (January 2009) 19:1–19:41

[42] Ligatti, J., Bauer, L., Walker, D.: Edit automata: enforcement mechanisms for run-time security policies. Int. J. Inf. Sec. '05

[43] Bauer, A., Leucker, M., Schallhart, C.: Monitoring of real-time properties. In Arun-Kumar, S., Garg, N., eds.: Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS). Volume 4337 of Lecture Notes in Computer Science., Berlin, Heidelberg, Springer-Verlag (December 2006)

[44] Bauer, A., Leucker, M., Schallhart, C.: The good, the bad, and the ugly, but how ugly is ugly? In Sokolsky, O., Tasiran, S., eds.: Proceedings of the 7th International Workshop on Runtime Verification (RV). Volume 4839 of Lecture Notes in Computer Science., Berlin, Heidelberg, Springer-Verlag (November 2007) 126–138

[45] Bauer, A., Leucker, M., Schallhart, C.: Comparing LTL semantics for runtime verification. Logic and Computation **20**(3) (2010) 651–674

[46] Bauer, A., Leucker, M., Schallhart, C.: Runtime verification for LTL and TLTL. ACM Trans. Softw. Eng. Methodol. **20**(4) (September 2011) 14:1–14:64

[47] Bauer, A., Falcone, Y.: Decentralised LTL monitoring. In Giannakopoulou, D., Méry, D., eds.: FM 2012: Formal Methods. Volume 7436 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2012) 85–100

[48] Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M., Reagle, J.M.: The platform for privacy preferences 1.0 (p3p1.0) specification. World Wide Web Consortium, Recommendation REC-P3P-20020416 (April 2002)

[49] Reagle, J., Cranor, L.F.: The platform for privacy preferences. Commun. ACM **42**(2) (February 1999) 48–55

[50] Ashley, P., Hada, S., Karjoth, G., Powers, C., Schunter, M.: Enterprise Privacy Authorization Language (EPAL). Technical report, IBM Research, Rüschlikon (2003)

[51] Karjoth, G., Schunter, M.: A privacy policy model for enterprises. In: Proceedings of the 15th IEEE Workshop on Computer Security Foundations. CSFW '02, Washington, DC, USA, IEEE Computer Society (2002) 271–281

[52] May, M.J., Gunter, C.A., Lee, I.: Privacy apis: Access control techniques to analyze and verify legal privacy policies. In: Proceedings of the 19th IEEE Workshop on Computer Security Foundations. CSFW '06, Washington, DC, USA, IEEE Computer Society (2006) 85–97

[53] Barth, A., Datta, A., Mitchell, J.C., Nissenbaum, H.: Privacy and contextual integrity: Framework and applications. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy. SP '06, Washington, DC, USA, IEEE Computer Society (2006) 184–198

[54] Barth, A., Mitchell, J., Datta, A., Sundaram, S.: Privacy and utility in business processes. In: Proceedings of the 20th IEEE Computer Security Foundations Symposium. CSF '07, Washington, DC, USA, IEEE Computer Society (2007) 279–294

[55] Dinesh, N., Joshi, A., Lee, I., Sokolsky, O.: Checking traces for regulatory conformance. In Leucker, M., ed.: Runtime Verification. Volume 5289 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2008) 86–103

[56] Lam, P., Mitchell, J., Sundaram, S.: A formalization of hipaa for a medical messaging system. In Fischer-Hübner, S., Lambrinoudakis, C., Pernul, G., eds.: Trust, Privacy and Security in Digital Business. Volume 5695 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2009) 73–85

[57] Ni, Q., Bertino, E., Lobo, J., Brodie, C., Karat, C.M., Karat, J., Trombeta, A.: Privacy-aware role-based access control. ACM Trans. Inf. Syst. Secur. **13**(3) (July 2010) 24:1–24:31

Appendix

A Temporal Mode Checking $\vdash_{\mathbf{B}}$ and $\vdash$ Judgements

In this section, we present the formal rules for the temporal mode checking $\vdash_{\mathbf{B}}$ and $\vdash$ judgement. The complete set of rules for $\vdash_{\mathbf{B}}$ judgements are shown in Figure 7.

The complete set of rules for $\vdash$ judgements are shown in Figures 8, 9, and 10. Note that, we do not present the cases of $\boxminus$, $\square$, $\ominus$, and $\bigcirc$. The mode checking judgements for $\boxminus$ and $\ominus$ are exactly like the $\diamondminus$ case. On the other hand, the mode checking judgements for $\square$ and $\bigcirc$ are exactly like the $\Diamond$ case.

B Definition of `ips`

We present the complete definition of **`ips`** in Figure 11.

C Updating Summary Structures

In this section, we present how to update the summary structure for the current trace position if we are given the summary structures for the previous trace position.

C.1 Summary Structure For $\ominus_{\mathbb{I}}\varphi$

We now explain how to incrementally maintain the structure for buildable temporal sub-formula of form $\ominus_{[lo,hi]}\varphi$. For each such formula, we have two summary structures $\mathbb{T}$ and $\mathbb{R}$. We denote the summary structures at execution position i as follows: $\mathbb{T}^i$ and $\mathbb{R}^i$. Each element of $\mathbb{T}^i$ is a substitution σ, which signifies that the formula φ was true with substitution σ at execution position i. Each element of $\mathbb{R}^i$ is a substitution σ which signifies that the formula $\ominus_{[lo,hi]}\varphi$ is true in the current execution position i with substitution σ. We now show how can we incrementally maintain the structure $\mathbb{T}^i$ and $\mathbb{R}^i$ provided that we have access to the structures $\mathbb{T}^{(i-1)}$ and $\mathbb{R}^{(i-1)}$.

$$\boxed{\boxed{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O}}$$

$$\frac{}{\chi_C \vdash_{\mathbf{B}} \top : \{\}}\;[\textit{\textbf{B-TRUE}}] \qquad \frac{}{\chi_C \vdash_{\mathbf{B}} \bot : \{\}}\;[\textit{\textbf{B-FALSE}}]$$

$$\frac{\forall k \in I(\mathsf{p}).fv(t_k) \subseteq \chi_C \quad \chi_O = \displaystyle\bigcup_{j \in O(\mathsf{p})} fv(t_j)}{\chi_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O}\;[\textit{\textbf{B-PRE}}]$$

$$\frac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_1 \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \, \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\;[\textit{\textbf{B-SINCE}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cup \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \wedge \varphi_2 : \chi_O}\;[\textit{\textbf{B-AND}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cap \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \vee \varphi_2 : \chi_O}\;[\textit{\textbf{B-OR}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1 \setminus \{\vec{x}\}}{\chi_C \vdash_{\mathbf{B}} \exists \vec{x}.\varphi : \chi_O}\;[\textit{\textbf{B-EXISTS}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad fv(\varphi_1), fv(\varphi_2) \subseteq \chi_C \cup \{\vec{x}\} \quad \{\vec{x}\} \subseteq \chi_1 \quad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2}{\chi_C \vdash_{\mathbf{B}} \forall \vec{x}.(\varphi_1 \to \varphi_2) : \{\}}\;[\textit{\textbf{B-UNIV}}]$$

$$\frac{\{\} \vdash_{\mathbf{B}} \varphi : \chi_O}{\chi_C \vdash_{\mathbf{B}} \boxminus_{\mathbb{I}} \varphi : \chi_O}\;[\textit{\textbf{B-HIST}}] \qquad \frac{\{\} \vdash_{\mathbf{B}} \varphi : \chi_O}{\chi_C \vdash_{\mathbf{B}} \diamondminus_{\mathbb{I}} \varphi : \chi_O}\;[\textit{\textbf{B-ONCE}}] \qquad \frac{\{\} \vdash_{\mathbf{B}} \varphi : \chi_O}{\chi_C \vdash_{\mathbf{B}} \ominus_{\mathbb{I}} \varphi : \chi_O}\;[\textit{\textbf{B-LAST}}]$$

Figure 7: Temporal mode checking $\vdash_{\mathbf{B}}$ judgements

$$\boxed{\boxed{\chi_C, \chi_F \vdash \varphi : \chi_O}}$$

$$\frac{}{\chi_C, \chi_F \vdash \top : \{\}}\;[\boldsymbol{\mathit{TRUE}}] \qquad \frac{}{\chi_C, \chi_F \vdash \bot : \{\}}\;[\boldsymbol{\mathit{FALSE}}]$$

$$\frac{\forall k \in I(\mathsf{p}).fv(t_k) \subseteq (\chi_C \cup \chi_F) \qquad \chi_O = \bigcup_{j \in O(\mathsf{p})} fv(t_j)}{\chi_C, \chi_F \vdash \mathsf{p}(t_1, \ldots, t_n) : \chi_O}\;[\boldsymbol{\mathit{PRE\text{-}1}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O}{\chi_C, \chi_F \vdash \mathsf{p}(t_1, \ldots, t_n) : \chi_O}\;[\boldsymbol{\mathit{PRE\text{-}2}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad \chi_C \cup \chi_1, \chi_F \vdash \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cup \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \wedge \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{AND\text{-}1}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cup \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \wedge \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{AND\text{-}2}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cup \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \wedge \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{AND\text{-}3}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cup \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \wedge \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{AND\text{-}4}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad \chi_C, \chi_F \vdash \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cap \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \vee \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{OR\text{-}1}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \chi_C, \chi_F \vdash \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cap \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \vee \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{OR\text{-}2}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \quad \chi_O = \chi_1 \cap \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \vee \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{OR\text{-}3}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cap \chi_2}{\chi_C, \chi_F \vdash \varphi_1 \vee \varphi_2 : \chi_O}\;[\boldsymbol{\mathit{OR\text{-}4}}]$$

Figure 8: Temporal mode checking $\vdash$ judgements (base cases and logical connective cases)

$$\boxed{\boxed{\chi_C, \chi_F \vdash \varphi : \chi_O}}$$

$$\frac{\chi_C, \chi_F \vdash \varphi : \chi_1 \quad \chi_O = \chi_1 \setminus \{\vec{x}\}}{\chi_C, \chi_F \vdash \exists \vec{x}.\varphi : \chi_O} \; [\textbf{\textit{EXIST-1}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1 \setminus \{\vec{x}\}}{\chi_C, \chi_F \vdash \exists \vec{x}.\varphi : \chi_O} \; [\textbf{\textit{EXIST-2}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \{\vec{x}\} \subseteq \chi_1 \quad fv(\varphi_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\} \quad fv(\varphi_2) \subseteq (\chi_C \cup \chi_1 \cup \chi_F) \quad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_2 : \chi_2}{\chi_C, \chi_F \vdash \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) : \{\}} \; [\textbf{\textit{UNIV}}$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad \{\vec{x}\} \subseteq \chi_1 \quad fv(\varphi_1) \subseteq \chi_C \cup \{\vec{x}\} \quad fv(\varphi_2) \subseteq (\chi_C \cup \chi_1 \cup \chi_F) \quad \chi_C \cup \chi_1, \chi_F \vdash \varphi_2 : \chi_2}{\chi_C, \chi_F \vdash \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) : \{\}} \; [\textbf{\textit{UNIV-2}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \quad \{\vec{x}\} \subseteq \chi_1 \quad fv(\varphi_1) \subseteq \chi_C \cup \{\vec{x}\} \quad fv(\varphi_2) \subseteq (\chi_C \cup \chi_1) \quad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2}{\chi_C, \chi_F \vdash \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) : \{\}} \; [\textbf{\textit{UNIV-3}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_1 : \chi_1 \quad \{\vec{x}\} \subseteq \chi_1 \quad fv(\varphi_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\} \quad fv(\varphi_2) \subseteq \chi_C \quad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2}{\chi_C, \chi_F \vdash \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2) : \{\}} \; [\textbf{\textit{UNIV-4}}]$$

Figure 9: Temporal mode checking $\vdash$ judgements (quantifier cases)

We have $\mathbb{T}^i \leftarrow \texttt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \varphi)$. Once we have updated $\mathbb{T}^i$, we can update $\mathbb{R}^i$ in the following way.

$$\mathbb{R}^i = \begin{cases} \emptyset \text{ when } i = 0 \\ \{\sigma \mid \sigma \in \mathbb{T}^{(i-1)} \wedge (lo \leq \tau_i - \tau_{(i-1)} \leq hi)\} \text{ when } i > 0 \end{cases}$$

C.2 Summary Structure For $\diamondminus_{\mathbb{I}}\varphi$

We now explain how to incrementally maintain the structure for buildable temporal sub-formula of form $\diamondminus_{[lo,hi]}\varphi$. For each such formula, we have two summary structures $\mathbb{P}$ and $\mathbb{R}$. We denote the summary structures at execution position i as follows: $\mathbb{P}^i$ and $\mathbb{R}^i$. Each element of $\mathbb{P}^i$ is a pair of form $\langle \sigma, k \rangle$ which signifies that the formula φ was true with substitution σ at execution position k. Each element of $\mathbb{R}^i$ is a substitution σ which signifies that the formula $\diamondminus_{[lo,hi]}\varphi$ is true in the current execution position i with substitution σ. We now show how can we incrementally maintain the structure $\mathbb{P}^i$ and $\mathbb{R}^i$ provided that we have access to the structures $\mathbb{P}^{(i-1)}$ and $\mathbb{R}^{(i-1)}$. Note that, we assume both $\mathbb{P}^{(-1)}$ and $\mathbb{R}^{(-1)}$ to be empty.

$$\begin{array}{l} \Sigma \leftarrow \texttt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \varphi) \\ S_a \leftarrow \{\langle \sigma, i \rangle | \langle \sigma, i \rangle \in \Sigma \wedge 0 \leq hi\} \\ S_r \leftarrow \{\langle \sigma, k \rangle | \langle \sigma, k \rangle \in \mathbb{P}^{(i-1)} \wedge \tau_i - \tau_k > hi\} \\ \mathbb{P}^i \leftarrow (\mathbb{P}^{(i-1)} \setminus S_r) \cup S_a \\ \mathbb{R}^i \leftarrow \{\sigma | \exists k.\langle \sigma, k \rangle \in \mathbb{P}^i \wedge \tau_i - \tau_k \in [lo, hi]\} \end{array}$$

The set Σ contains all the substitutions for which φ holds true in execution position i. The set S_a contains all the new pairs of $\langle \sigma, i \rangle$ denoting that φ holds with substitution σ at i. The set

$$\boxed{\boxed{\chi_C, \chi_F \vdash \varphi : \chi_O}}$$

$$\frac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \quad \chi_1, (\chi_C \cup \chi_F) \vdash \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{SINCE-1}}]$$

$$\frac{\{\}, \chi_C \cup \chi_F \vdash \varphi_2 : \chi_1 \quad \{\}, \chi_C \cup \chi_F \cup \chi_1 \vdash \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{SINCE-2}}]$$

$$\frac{\{\}, \chi_C \cup \chi_F \vdash \varphi_2 : \chi_1 \quad \{\} \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{SINCE-3}}]$$

$$\frac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \quad \chi_1 \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{SINCE-4}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \quad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{UNTIL-1}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \quad \chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{UNTIL-2}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_2 : \chi_1 \quad \chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{UNTIL-3}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi_2 : \chi_1 \quad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}\ [\textbf{\textit{UNTIL-4}}]$$

$$\frac{\{\}, \chi_C \cup \chi_F \vdash \varphi : \chi_1 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \diamondminus_{\mathbb{I}} \varphi : \chi_O}\ [\textbf{\textit{ONCE-1}}]$$

$$\frac{\{\} \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \diamondminus_{\mathbb{I}} \varphi : \chi_O}\ [\textbf{\textit{ONCE-2}}]$$

$$\frac{\chi_C, \chi_F \vdash \varphi : \chi_1 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \Diamond_{\mathbb{I}} \varphi : \chi_O}\ [\textbf{\textit{EVENTUALLY-1}}]$$

$$\frac{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \Diamond_{\mathbb{I}} \varphi : \chi_O}\ [\textbf{\textit{EVENTUALLY-2}}]$$

Figure 10: Temporal mode checking $\vdash$ judgements (temporal operator cases)

$$
\begin{aligned}
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \top) &= \{\sigma_{\text{in}}\} \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \bot) &= \{\} \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \mathsf{p}(t_1, \ldots, t_n)) &= \mathbf{sat}(\mathcal{L}, i, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}}) \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \vee \varphi_2) &= \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1) \bigcup \mathbf{ips}(\mathcal{L}, i, \pi, \sigma_{\text{in}}, \varphi_2) \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \wedge \varphi_2) &= \textstyle\bigcup_{\sigma_c \in \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1)} \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_c, \varphi_2) \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \exists \vec{x}.\varphi) &= \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi) \setminus \{\vec{x}\}
\end{aligned}
$$

$$
\begin{array}{l}
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \\
\forall \vec{x}.(\varphi_1 \rightarrow \varphi_2))
\end{array}
\quad = \quad
\begin{array}{ll}
\text{let} & \Sigma_1 \leftarrow \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1) \\
\text{return} & \begin{cases} \{\} & \text{if } \exists \sigma_c \in \Sigma_1.(\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_c, \varphi_2) = \{\}) \\ \{\sigma_{\text{in}}\} & \text{otherwise} \end{cases}
\end{array}
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \alpha\, \mathcal{S}_{\mathbb{I}} \beta) = \sigma_{\text{in}} \bowtie \pi.\mathcal{A}(\alpha\, \mathcal{S}_{\mathbb{I}} \beta)(i).\mathbb{R} \text{ if } \mathbf{B} \in \mathit{label}(\alpha\, \mathcal{S}_{\mathbb{I}} \beta)
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \alpha\, \mathcal{S}_{\mathbb{I}} \beta) \quad = \quad
\begin{array}{l}
\text{If } \mathbf{B} \notin \mathit{label}(\alpha\, \mathcal{S}_{\mathbb{I}} \beta) \\
\text{let} \quad S_\beta \leftarrow \{\langle \sigma, k \rangle | k = \max l.((0 \leq l \leq i) \wedge ((\tau_i - \tau_l) \in \mathbb{I}) \\
\qquad\qquad \wedge \sigma \in \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{\text{in}}, \beta))\} \\
\qquad S_{R_1} \leftarrow \{\sigma | \langle \sigma, i \rangle \in S_\beta \wedge 0 \in \mathbb{I}\} \\
\qquad S_{R_2} \leftarrow \{\bowtie \sigma_l^\alpha \neq \sigma_\bot | \exists \langle \sigma_\beta, k \rangle \in S_\beta . k < i \wedge \\
\qquad\qquad \forall l.(k < l \leq i \rightarrow \sigma_l^\alpha \in \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \varphi_1))\} \\
\text{return } S_{R_1} \cup S_{R_2}
\end{array}
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \ominus_{\mathbb{I}} \alpha) =
\begin{cases}
\sigma_{in} \bowtie \pi.\mathcal{A}(\ominus_{\mathbb{I}} \alpha)(i).\mathbb{R} & \text{if } \mathbf{B} \in \mathit{label}(\ominus_{\mathbb{I}} \alpha) \\
\mathbf{ips}(\mathcal{L}, i-1, \tau, \pi, \sigma_{in}, \alpha) & \text{if } i > 1, \tau_i - \tau_{(i-1)} \in \mathbb{I}, \text{ and } \mathbf{B} \notin \mathit{label}(\ominus_{\mathbb{I}} \alpha) \\
\{\} & \text{otherwise}
\end{cases}
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \diamondminus_{\mathbb{I}} \alpha) =
\begin{cases}
\sigma_{in} \bowtie \pi.\mathcal{A}(\diamondminus_{\mathbb{I}} \alpha)(i).\mathbb{R} & \text{if } \mathbf{B} \in \mathit{label}(\diamondminus_{\mathbb{I}} \alpha) \\
\{\sigma \mid \exists k.(k \leq i \wedge \tau_i - \tau_k \in \mathbb{I} \wedge \sigma \in \mathbf{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \alpha))\} & \text{if } \mathbf{B} \notin \mathit{label}(\diamondminus_{\mathbb{I}} \alpha)
\end{cases}
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \boxminus_{\mathbb{I}} \alpha) =
\begin{cases}
\sigma_{in} \bowtie \pi.\mathcal{A}(\boxminus_{\mathbb{I}} \alpha)(i).\mathbb{R} & \text{if } \mathbf{B} \in \mathit{label}(\boxminus_{\mathbb{I}} \alpha) \\
\{\sigma \mid \forall k.(0 \leq k \leq i \wedge \tau_i - \tau_k \in \mathbb{I} \wedge \sigma \in \mathbf{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \alpha))\} & \text{if } \mathbf{B} \notin \mathit{label}(\boxminus_{\mathbb{I}} \alpha)
\end{cases}
$$

$$
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \alpha\, \mathcal{U}_{\mathbb{I}} \beta) \quad = \quad
\begin{array}{l}
\text{let} \quad S_\beta \leftarrow \{\langle \sigma, k \rangle | k = \min l.(l \geq i \wedge ((\tau_l - \tau_i) \in \mathbb{I}) \\
\qquad\qquad \wedge \sigma \in \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{\text{in}}, \beta))\} \\
\qquad S_{R_1} \leftarrow \{\sigma | \langle \sigma, i \rangle \in S_\beta \wedge 0 \in \mathbb{I}\} \\
\qquad S_{R_2} \leftarrow \{\bowtie \sigma_l^\alpha \neq \sigma_\bot | \exists \langle \sigma_\beta, k \rangle \in S_\beta . k \neq i \wedge \\
\qquad\qquad \forall (i \leq l < k). \sigma_l^\alpha \in \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \alpha)\} \\
\text{return } S_{R_1} \cup S_{R_2}
\end{array}
$$

$$
\begin{aligned}
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \bigcirc_{\mathbb{I}} \alpha) &= \{\sigma \mid \sigma \in \mathbf{ips}(\mathcal{L}, i+1, \tau, \pi, \sigma_{in}, \alpha) \wedge (\tau_{(i+1)} - \tau_i \in \mathbb{I})\} \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \Diamond_{\mathbb{I}} \alpha) &= \{\sigma \mid \exists k.(k \geq i \wedge \tau_k - \tau_i \in \mathbb{I} \wedge \sigma \in \mathbf{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \alpha))\} \\
\mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \Box_{\mathbb{I}} \alpha) &= \{\sigma \mid \forall k.(k \geq i \wedge \tau_k - \tau_i \in \mathbb{I} \wedge \sigma \in \mathbf{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \alpha))\}
\end{aligned}
$$

Figure 11: The definition of the **ips** function.

S_r contains all the pairs of form $\langle \sigma, k \rangle$ where φ was true with substitution σ in k and it violates the interval constraint $[lo, hi]$. Thus, we add the new pairs and throw out the old pairs from the structure $\mathbb{P}^{(i-1)}$ to get the new structure $\mathbb{P}^i$. Once we have updated the structure $\mathbb{P}^i$, we calculate the structure $\mathbb{R}^i$ by choosing σ out of pairs $\langle \sigma, k\rangle \in \mathbb{P}^i$ for which the interval constraint is satisfied.

C.3 Summary Structure For $\boxminus_{\mathbb{I}}\varphi$

In this section, we explain how to incrementally maintain the structure for buildable temporal sub-formula of form $\boxminus_{[lo,hi]}\varphi$. For each such formula, we have two summary structures $\mathbb{H}$ and $\mathbb{R}$. We denote the summary structures at execution position i as follows: $\mathbb{H}^i$ and $\mathbb{R}^i$. Each element of $\mathbb{H}^i$ is a triple of form $\langle \sigma, l, r\rangle$ which signifies that the formula φ was true with substitution σ from execution position l to r, inclusive. Each element of $\mathbb{R}^i$ is a substitution σ which signifies that the formula $\boxminus_{[lo,hi]}\varphi$ is true in the current execution position i with substitution σ. We now show how can we incrementally maintain the structure $\mathbb{H}^i$ and $\mathbb{R}^i$ provided that we have access to the structures $\mathbb{H}^{(i-1)}$ and $\mathbb{R}^{(i-1)}$. Note that, we assume both $\mathbb{H}^{(-1)}$ and $\mathbb{R}^{(-1)}$ to be empty.

In our construction, Σ denotes the set of substitutions for which φ holds in the current execution position i. The construction first checks to see whether a current substitution can extend the range of an existing triple from $\langle \sigma, l, i-1\rangle \in \mathbb{H}^{(i-1)}$ to $\langle \sigma', l, i\rangle$ where $\sigma' \geq \sigma$. If yes, those extended tuples are added in the set S_{update}. Then we add each substitution which does not extend an existing triple, to start a new triple $\langle \sigma, i, i\rangle$ which signifies that φ holds for σ in position i. They are added to the set S_{new}. Next we get all existing triples which cannot be extended with any new substitution. They are added to the set $S_{\text{carry-over}}$. Finally, we throw out those triples whose right end does not satisfy the interval constraint. They are stored in the set S_{remove}. We then add all the triples either in S_{new}, S_{update}, or $S_{\text{carry-over}}$ and remove all the triples in S_{remove}. The result is stored in $\mathbb{H}^i$. Once $\mathbb{H}^i$ has been calculated, we then show how to calculate the result set $\mathbb{R}^i$.

$$
\begin{array}{lll}
\Sigma & \leftarrow & \texttt{\textbf{ips}}(\mathcal{L}, i, \tau, \pi, \bullet, \varphi) \\
S_{\text{new}} & \leftarrow & \{\langle \sigma, i, i\rangle \mid \sigma \in \Sigma \wedge \forall \langle \sigma_1, l, i-1\rangle \in \mathbb{H}^{(i-1)}.\sigma \bowtie \sigma_1 \neq \sigma\} \\
S_{\text{update}} & \leftarrow & \{\langle \sigma_1 \bowtie \sigma_2, l, i\rangle \mid \sigma_1 \in \Sigma \wedge \langle \sigma_2, l, i-1\rangle \in \mathbb{H}^{(i-1)} \wedge \sigma_1 \bowtie \sigma_2 \neq \sigma_\perp\} \\
S_{\text{carry-over}} & \leftarrow & \{\langle \sigma, l, r\rangle \mid \langle \sigma, l, r\rangle \in \mathbb{H}^{(i-1)} \wedge (r < (i-1) \vee \\
 & & \qquad (r = i-1 \wedge \forall \sigma_1 \in \Sigma.\sigma \bowtie \sigma_1 \neq \sigma))\} \\
 & & \\
T & \leftarrow & S_{\text{new}} \bigcup S_{\text{update}} \bigcup S_{\text{carry-over}} \\
 & & \\
S_{\text{remove}} & \leftarrow & \{\langle \sigma, l, r\rangle \mid \langle \sigma, l, r\rangle \in T \wedge (\tau_i - \tau_r) > hi\} \\
\mathbb{H}^i & \leftarrow & T \setminus S_{\text{remove}} \\
tl & \leftarrow & \texttt{\textbf{minPosition}}(\tau, i, lo, hi) \\
th & \leftarrow & \texttt{\textbf{maxPosition}}(\tau, i, lo, hi) \\
\mathbb{R}^i & \leftarrow & \{\sigma \mid tl \neq -1 \wedge th \neq -1 \wedge \exists l, r.(\langle \sigma, l, r\rangle \in \mathbb{H}^i \wedge (l \leq tl \leq th \leq r))\}
\end{array}
$$

For calculating $\mathbb{R}^i$, we use two auxiliary utility functions, **minPosition** and **maxPosition**. The function **minPosition** (resp., **maxPosition**) takes as input the time stamp sequence τ, the current trace position i, the lower bound of the interval lo and the upper bound of the interval hi. The function **minPosition** (resp., **maxPosition**) returns the minimum (resp., maximum) position tp such that it satisfies $0 \leq tp \leq i$ and $lo \leq \tau_i - \tau_{tp} \leq hi$. If such positions are not found, both functions return -1.

Let us consider tl to be the result of the **minPosition** function whereas th to be the result of the **maxPosition** function. To calculate the substitutions in $\mathbb{R}^i$, we choose the triples $\langle \sigma, l, r\rangle$ in

$\mathbb{H}^i$ that satisfies the following constraint: $(l \leq tl \leq th \leq r)$.

C.4 Summary Structure For α in $\alpha \mathcal{S}_{\mathbb{I}} \beta$

In this section, we will show how the structure $\mathbb{S}_\alpha$ at i is updated. We first use **ips** to calculate a set of substitutions (Σ_α) that satisfy the following constraints: every substitution σ' in Σ_α (1) extends a substitution σ, which makes β true at a previous time point k and (2) makes $\alpha\sigma'$ true at position i. Next, we identify all substitutions σ such that $\alpha\sigma$ is true at position i and it is not the case that $\alpha\sigma$ has been true since an earlier time point k till i. In other words, i is the first position, from which till the current time point, $\alpha\sigma$ has been true. We store them in the set S_{new}. Then we collect pairs $\langle\sigma_a, k\rangle$ from $\mathbb{S}_\alpha^{(i-1)}$ such that α holds true at i. We then compute the join of the substitutions with which α holds true in the current state with σ_a. These pairs are stored in S_{update}. Computing the join ensures that α has been true with the same substitution. In the case where $\alpha(x, y) = A(x^-) \mathcal{S}_{\mathbb{I}} B(y^-)$, the structure for α needs to record substitutions for both x and y, even though sometimes only substitutions for y is available. If we omit substitutions for x, we could run into situations where $\alpha(1, 2)$ is true at i, $\alpha(2, 2)$ at $i + 1$, and we mistakenly think that $\alpha(x, y)$ has been true from i to $i + 1$ with y instantiated to 2. Recording substitutions for both x and y, and taking a join will rule out this case. Finally the new summary structure $\mathbb{S}_\alpha^i$ is the union of the two sets. Note that, we assume $\mathbb{S}_\alpha^{(-1)}$ to be empty.

$$
\begin{array}{lll}
\Sigma_\alpha & \leftarrow & \displaystyle\bigcup_{\langle\sigma,k\rangle \in \mathbb{S}_\beta^i \wedge k \neq i} \textbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma, \alpha) \\
S_{\text{new}} & \leftarrow & \{\langle\sigma, i\rangle \mid \sigma \in \Sigma_\alpha \wedge \forall\langle\sigma_a, k\rangle \in \mathbb{S}_\alpha^{(i-1)}.(\sigma \bowtie \sigma_a \neq \sigma)\} \\
S_{\text{update}} & \leftarrow & \{\langle\sigma \bowtie \sigma_a, k\rangle \mid \langle\sigma_a, k\rangle \in \mathbb{S}_\alpha^{(i-1)} \wedge \sigma \in \Sigma_\alpha \wedge \sigma \bowtie \sigma_a \neq \sigma_\perp\} \\
\mathbb{S}_\alpha^i & \leftarrow & S_{\text{new}} \cup S_{\text{update}}
\end{array}
$$

D Correctness of **précis**

In this section, we prove the correctness of our algorithm **précis** (Theorem 1). However, we first prove some auxiliary lemmas which will be used to prove the Theorem 1. We also introduce the readers with our different data structures (state π).

D.1 Properties of $\vdash$ and $\vdash_{\mathbf{B}}$ Judgement

We start by defining what it means for a policy formula φ to be well-moded and then define when do we call φ a **B-formula**.

Definition 1 (Well-moded formulas). *A formula φ is* well-moded *with respect to a given χ_C and χ_F if we can derive the following judgement for φ: $\chi_C, \chi_F \vdash \varphi : \chi_O$ where $\chi_O \subseteq fv(\varphi)$.*

Definition 2 (**B-formula**). *Given χ_C and χ_F, for all formulas φ such that $\chi_C, \chi_F \vdash \varphi : \chi_O$ and $\chi_O \subseteq fv(\varphi)$, we say φ is a **B-formula** (or, $\boldsymbol{B} \in label(\varphi)$) iff the following judgement can be derived for φ: $\chi_C \vdash_{\boldsymbol{B}} \varphi : \chi_O{}'$ where $\chi_O{}' \subseteq fv(\varphi)$. In the same vein, if a formula φ does not satisfy the above, we write $\boldsymbol{B} \notin label(\varphi)$ or φ is not a **B-formula**.*

Lemma 1 (Upper Bound of $\vdash_{\mathbf{B}}$). *For all φ, χ_C and χ_O, if $\chi_C \vdash_{\boldsymbol{B}} \varphi : \chi_O$, then $\chi_O \subseteq fv(\varphi)$.*

Proof. Induction on the derivation of $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$.

Cases [B-TRUE], [B-FALSE].

Then $\varphi = \top$ or $\varphi = \bot$, $\chi_O = \{\}$ and $fv(\varphi) = \{\}$. Trivially $\chi_O \subseteq fv(\varphi)$.

Case [B-PRE].

Then $\varphi = \mathsf{p}(t_1, \ldots, t_n)$, $\chi_O = \bigcup_{j \in O(\mathsf{p})} fv(t_j)$ by [B-PRE] and $fv(\mathsf{p}(t_1, \ldots, t_n)) = \bigcup_{j \in \{1,\ldots,n\}} fv(t_j)$ by definition. Thus trivially $\chi_O \subseteq fv(\varphi)$.

Case [B-SINCE].

Then $\varphi = \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$ and $\dfrac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_1 \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}$. By inductive hypothesis, $\chi_1 \subseteq fv(\varphi_2)$. Thus $\chi_O = \chi_1 \subseteq fv(\varphi_2) \subseteq fv(\varphi_1) \cup fv(\varphi_2) = fv(\varphi)$.

Case [B-AND].

Then $\varphi = \varphi_1 \wedge \varphi_2$ and $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cup \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \wedge \varphi_2 : \chi_O}$. By inductive hypothesis, $\chi_1 \subseteq fv(\varphi_1)$ and $\chi_2 \subseteq fv(\varphi_2)$. Thus $\chi_O = \chi_1 \cup \chi_2 \subseteq fv(\varphi_1) \cup fv(\varphi_2) = fv(\varphi)$.

Case [B-OR].

Then $\varphi = \varphi_1 \vee \varphi_2$ and $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cap \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \vee \varphi_2 : \chi_O}$. By inductive hypothesis, $\chi_1 \subseteq fv(\varphi_1)$ and $\chi_2 \subseteq fv(\varphi_2)$. Thus $\chi_O = \chi_1 \cap \chi_2 \subseteq fv(\varphi_1) \cap fv(\varphi_2) \subseteq fv(\varphi_1) \cup fv(\varphi_2) = fv(\varphi)$.

Case [B-EXISTS].

Then $\varphi = \exists \vec{x}.\varphi$ and $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1 \setminus \{\vec{x}\}}{\chi_C \vdash_{\mathbf{B}} \exists \vec{x}.\varphi : \chi_O}$. By inductive hypothesis, $\chi_1 \subseteq fv(\varphi_1)$. By set properties, $\chi_O = \chi_1 \setminus \{\vec{x}\} \subseteq fv(\varphi_1) \setminus \{\vec{x}\} = fv(\exists \vec{x}.\varphi)$.

□

Lemma 2 (Upper Bound of $\vdash$)**.** *For all φ, χ_C, χ_F and χ_O, if $\chi_C, \chi_F \vdash \varphi : \chi_O$, then $\chi_O \subseteq fv(\varphi)$.*

Proof. Induction on the derivation of $\chi_C, \chi_F \vdash \varphi : \chi_O$. Most cases are equivalent to Lemma 1. We again show select cases.

Case [UNIV-3], [UNIV-2], [UNIV-1], [UNIV-4].

Then $\chi_O = \{\}$, which is trivially a subset of the free variables of any formula.

Case [UNTIL-1].

Then $\varphi = \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2$ and $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \quad \chi_C, \chi_F \cup \chi_1 \vdash \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}$. By Lemma 1, $\chi_1 \subseteq fv(\varphi_2)$. Then $\chi_O = \chi_1 \subseteq fv(\varphi_2) \subseteq fv(\varphi_1) \cup fv(\varphi_2) = fv(\varphi)$.

Case [UNTIL-3].

Then $\varphi = \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2$ and $\dfrac{\chi_C, \chi_F \vdash \varphi_2 : \chi_1 \quad \chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \quad \chi_O = \chi_1}{\chi_C, \chi_F \vdash \varphi_1 \mathcal{U}_{\mathbb{I}} \varphi_2 : \chi_O}$. By inductive hypothesis, $\chi_1 \subseteq fv(\varphi_2)$. Then $\chi_O = \chi_1 \subseteq fv(\varphi_2) \subseteq fv(\varphi_1) \cup fv(\varphi_2) = fv(\varphi)$.

□

Lemma 3 (Buildable temporal subformula)**.** *For a given χ_C and a formula φ, if $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$ holds, then for all sub-formula $\hat{\varphi}$ of φ, there exists a χ'_C for which the judgment $\chi'_C \vdash_{\mathbf{B}} \hat{\varphi} : \chi'_O$ holds.*

Proof. The proof proceeds by doing an induction on the derivation of the $\vdash_{\mathbf{B}}$ judgements. □

Lemma 4 (Δ of buildable temporal formula)**.** *For all formula φ such that there exists a χ_C for which $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$ holds, then $\Delta(\varphi) = 0$.*

Proof. The proof proceeds by doing an induction on the structure of φ and case analysis of the Δ function. □

Lemma 5 (Monotonicity of $\vdash_{\mathbf{B}}$ judgement)**.** *For a given χ_C and a formula φ, if $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$ can be derived, then for any χ'_C such that $\chi'_C \supseteq \chi_C$, $\chi'_C \vdash_{\mathbf{B}} \varphi : \chi_O$ can be derived.*

Proof. We do induction on the derivation of the $\vdash_{\mathbf{B}}$ judgements. We show select cases and the other cases are similar.

Cases [B-TRUE], [B-FALSE].

We can see from the derivation of $\overline{\chi_C \vdash_{\mathbf{B}} \top : \{\}}$ and $\overline{\chi_C \vdash_{\mathbf{B}} \bot : \{\}}$ that the premise of the judgements do not use χ_C, thus we can trivially write $\chi'_C \vdash_{\mathbf{B}} \top : \emptyset$ and $\chi'_C \vdash_{\mathbf{B}} \bot : \emptyset$, without changing the derivation.

Case [B-PRE].

From the first premise of the judgement, it is required that $\forall k \in I(\mathsf{p}).fv(t_k) \subseteq \chi_C$. We know $\chi'_C \supseteq \chi_C$. Thus, we can write $\forall k \in I(\mathsf{p}).fv(t_k) \subseteq \chi'_C$. Then we get the judgement $\chi'_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O$.

Case [B-SINCE].

Then $\dfrac{\{\} \vdash_{\mathbf{B}} \varphi_2 : \chi_1 \qquad \chi_1 \vdash_{\mathbf{B}} \varphi_1 : \chi_2 \qquad \chi_O = \chi_1}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O}$. We can see that the premises do not use χ_C. Thus, we can replace χ_C with χ'_C and can derive the judgement $\chi'_C \vdash_{\mathbf{B}} \varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2 : \chi_O$.

Case [B-AND].

Then $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cup \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \wedge \varphi_2 : \chi_O}$. We see that it is required that $\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1$ and $\chi_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2$. From I.H., we can write $\chi'_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1$ and $\chi'_C \cup \chi_1 \vdash_{\mathbf{B}} \varphi_2 : \chi_2$ as $(\chi'_C \cup \chi_1) \supseteq (\chi_C \cup \chi_1)$. Thus, enabling us to derive the judgement $\chi'_C \vdash_{\mathbf{B}} \varphi_1 \wedge \varphi_2 : \chi_O$.

Case [B-OR].

Then $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1 \qquad \chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2 \qquad \chi_O = \chi_1 \cap \chi_2}{\chi_C \vdash_{\mathbf{B}} \varphi_1 \vee \varphi_2 : \chi_O}$. We see that it is required that $\chi_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1$ and $\chi_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2$. From I.H., we can write $\chi'_C \vdash_{\mathbf{B}} \varphi_1 : \chi_1$ and $\chi'_C \vdash_{\mathbf{B}} \varphi_2 : \chi_2$. Thus, enabling us to derive the judgement $\chi'_C \vdash_{\mathbf{B}} \varphi_1 \vee \varphi_2 : \chi_O$.

Case [B-EXISTS].

Then $\dfrac{\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1 \quad \chi_O = \chi_1 \setminus \{\vec{x}\}}{\chi_C \vdash_{\mathbf{B}} \exists\vec{x}.\varphi : \chi_O}$. We see that it is required that $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_1$. From I.H., we can write $\chi'_C \vdash_{\mathbf{B}} \varphi : \chi_1$. Thus, enabling us to derive the judgment $\chi'_C \vdash_{\mathbf{B}}: \exists\vec{x}.\varphi\chi_O$.

□

Lemma 6 (Invariance of $\vdash_{\mathbf{B}}$)**.** *Given χ_C and χ_F, for all formulas φ of form $\varphi_1 \,\mathcal{S}_{\mathbb{I}}\varphi_2$, $\diamondminus_{\mathbb{I}}\varphi$, $\boxminus_{\mathbb{I}}\varphi$, or $\ominus_{\mathbb{I}}\varphi$, if $\chi_C \vdash_{\boldsymbol{B}} \varphi : \chi'_O$ can be derived then $\{\} \vdash_{\boldsymbol{B}} \varphi : \chi'_O$ can be derived.*

Proof. The proof follows from the judgements [B-SINCE], [B-ONCE], [B-HIST], and [B-LAST], respectively, in Figure 7. None of the premises of the judgements [B-SINCE], [B-ONCE], [B-HIST], and [B-LAST] use χ_C. We can thus replace χ_C with $\{\}$ without changing the judgement result. □

Lemma 7 (Invariance of $\vdash$)**.** *For all formula φ and for some given χ_C, if $\chi_C \vdash_{\boldsymbol{B}} \varphi : \chi_O$ holds then $\chi_C, \{\} \vdash \varphi : \chi_O$ holds.*

Proof. Induction on the derivation of the $\vdash_{\mathbf{B}}$ and $\vdash$ judgements. □

Lemma 8 (Monotonicity of $\vdash$)**.** *Given χ_C, χ_F, and a formula φ, if $\chi_C, \chi_F \vdash \varphi : \chi_O$ can be derived, then for any χ'_C, χ'_F such that $\chi_C \subseteq \chi'_C$ and $\chi_F \subseteq \chi'_F$, the judgement $\chi'_C, \chi'_F \vdash \varphi : \chi_O$ can be derived.*

Proof. By induction on the derivation of the $\vdash$ judgements. We show select cases.

Case [TRUE]/[FALSE].

The premise of the judgement neither uses χ_C nor χ_F. Thus, we can write $\chi'_C, \chi'_F \vdash \top : \emptyset$ and $\chi'_C, \chi'_F \vdash \bot : \emptyset$.

Case [PRE-2].

From the derivation of the judgement, we see that it is required that $\chi_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O$. In this premise of the judgement, χ_F is not used. So, we can easily replace χ_F with χ'_F. By Lemma 5, if we have $\chi_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O$, we can write $\chi'_C \vdash_{\mathbf{B}} \mathsf{p}(t_1, \ldots, t_n) : \chi_O$. Thus, we can derive the judgement $\chi'_C, \chi'_F \vdash \mathsf{p}(t_1, \ldots, t_n) : \chi_O$.

Case [PRE-1].

From the derivation of the judgement, we see that it is required to satisfy $\bigcup_{k \in I(\mathsf{p})} .fv(t_k) \subseteq (\chi_C \cup \chi_F)$. As $\chi_C \subseteq \chi'_C$ and $\chi_F \subseteq \chi'_F$, we have $(\chi_C \cup \chi_F) \subseteq (\chi'_C \cup \chi'_F)$. Thus, we have $\chi'_C, \chi'_F \vdash \mathsf{p}(t_1, \ldots, t_n) : \chi_O$.

Case [SINCE-1]

From the derivation of the judgement, we see that it is required to satisfy $\emptyset \vdash_{\mathbf{B}} \varphi : \chi_1$ and $\chi_1, (\chi_C \cup \chi_F) \vdash \varphi_1 : \chi_2$. The first premise of the derivation of the judgement neither uses χ_C nor χ_F. However, this is not the case for the second premise. We can write $(\chi_C \cup \chi_F) \subseteq (\chi'_C \cup \chi'_F)$ as $\chi_C \subseteq \chi'_C$ and $\chi_F \subseteq \chi'_F$. From I.H., we have $\chi_1, (\chi'_C \cup \chi'_F) \vdash \varphi_1 : \chi_2$. We can thus derive the judgement $\chi'_C, \chi'_F \vdash \varphi_1 \,\mathcal{S}_{\mathbb{I}}\varphi_2 : \chi_O$.

The other cases are similar. □

Lemma 9 (Switching to $\vdash$ judgements from $\vdash_{\mathbf{B}}$ judgements)**.** *For a given χ_C and a formula φ, if $\chi_C \vdash_{\boldsymbol{B}} \varphi : \chi_O$ can be derived, then for any χ_F, $\chi_C, \chi_F \vdash \varphi : \chi_O$ can be derived.*

Proof. The proof follows from Lemma 7 and 8. According to Lemma 7, if we have $\chi_C \vdash_{\mathbf{B}} \varphi : \chi_O$, we can write $\chi_C, \emptyset \vdash \varphi : \chi_O$. By Lemma 8, if we have $\chi_C, \emptyset \vdash \varphi : \chi_O$, we can write $\chi_C, \chi_F \vdash \varphi : \chi_O$ for any χ_F as $\chi_F \supseteq \emptyset$. □

D.2 Substitutions and its Properties

The next notion necessary to understand our compliance checking algorithm is the notion of *substitution.* A substitution can be viewed as a finite mapping that maps free variables to concrete values in the domain $\mathcal{D}$. More formally, we define a substitution (denoted by σ) to be a finite mapping from variables to values in the domain $\mathcal{D}$, where $\sigma(v)$ is in the domain of the variable v. Given a substitution σ, $\textbf{\textsf{dom}}(\sigma)$ is defined as follows: $\textbf{\textsf{dom}}(\sigma) = \{x \mid \sigma(x) \neq x\}$. We use Σ (possibly with subscript or superscript) to denote a set of substitutions. We use $\bullet$ to represent the identity substitution and $\sigma_\perp$ to represent an invalid substitution.

We now define what it means for a substitution σ_1 to extend σ_2 (denoted $\sigma_1 \geq \sigma_2$). We also define how to apply a substitution σ to a formula φ with free variables.

Definition 3 (Extension of Substitution)**.** *Given two substitutions σ and σ', we say σ' extends σ, denoted by $\sigma' \geq \sigma$, if the following holds: $\textbf{\textsf{dom}}(\sigma') \supseteq \textbf{\textsf{dom}}(\sigma)$ and $\forall x \in \textbf{\textsf{dom}}(\sigma).(\sigma(x) = \sigma'(x))$.*

Definition 4 (Substitution Application)**.** *The application of a substitution σ to a formula φ, denoted $\varphi\sigma$, is recursively defined by*

$$\varphi\sigma = \begin{cases} \top & \varphi = \top \\ \bot & \varphi = \bot \\ \mathsf{p}(\sigma(t_1), \ldots, \sigma(t_n)) & \varphi = \mathsf{p}(t_1, \ldots, t_n) \\ (\varphi_1\sigma) \vee (\varphi_2\sigma) & \varphi = \varphi_1 \vee \varphi_2 \\ (\varphi_1\sigma) \wedge (\varphi_2\sigma) & \varphi = \varphi_1 \wedge \varphi_2 \\ \exists\vec{x}.\varphi[\sigma \setminus \{\vec{x}\}] & \varphi = \exists\vec{x}.\varphi \\ \forall\vec{x}.(\varphi_1[\sigma \setminus \{\vec{x}\}] \rightarrow \varphi_2[\sigma \setminus \{\vec{x}\}]) & \varphi = \forall\vec{x}.(\varphi_1 \rightarrow \varphi_2) \\ (\varphi_1\sigma)\, \mathcal{S}_{[c,d]} (\varphi_2\sigma) & \varphi = \varphi_1\, \mathcal{S}_{[c,d]} \varphi_2 \\ (\varphi_1\sigma)\, \mathcal{U}_{[c,d]} (\varphi_2\sigma) & \varphi = \varphi_1\, \mathcal{U}_{[c,d]} \varphi_2 \\ \diamondminus_{[c,d]}(\varphi\sigma) \mid \boxminus_{[c,d]}(\varphi\sigma) \mid \ominus_{[c,d]}(\varphi\sigma) & \varphi = \diamondminus_{[c,d]}\varphi \mid \boxminus_{[c,d]}\varphi \mid \ominus_{[c,d]}\varphi \\ \Diamond_{[c,d]}(\varphi\sigma) \mid \Box_{[c,d]}(\varphi\sigma) \mid \bigcirc_{[c,d]}(\varphi\sigma) & \varphi = \Diamond_{[c,d]}\varphi \mid \Box_{[c,d]}\varphi \mid \bigcirc_{[c,d]}\varphi \end{cases}$$

where σ is extended such that $\sigma(e) = e$ for any e not a variable or in the domain of σ.

We now introduce the readers with some notations we use. Given a substitution σ, we use $\sigma \downarrow S$ to denote a new substitution which is same as σ except all the variable, value mappings for variables *not* in set S are removed in the new substitution. Let $\sigma' = \sigma \downarrow S$, then the following holds: $\textbf{\textsf{dom}}(\sigma') \subseteq \textbf{\textsf{dom}}(\sigma)$, $\forall x \in S.(\sigma(x) = \sigma'(x))$, and $\forall x \in \textbf{\textsf{dom}}(\sigma).(x \notin S \rightarrow (\sigma'(x) = x))$. We now generalize the above operation for a set of substitutions. Consider Σ' is a set of substitutions and $\Sigma = \Sigma' \downarrow S$. We define $\Sigma = \Sigma' \downarrow S$ in the following way, $\forall \sigma \in \Sigma'.(\Sigma \leftarrow \Sigma \cup \{\sigma \downarrow S\})$.

We use $\sigma \backslash S$ to denote a new substitution which is same as σ except the variable, value mappings for variables in set S are removed. More precisely, consider $\sigma' = \sigma \setminus S$, then $\textbf{dom}(\sigma') \subseteq \textbf{dom}(\sigma)$, $\forall x \in \textbf{dom}(\sigma).(x \notin S \rightarrow (\sigma(x) = \sigma'(x)))$, and $\forall x \in \textbf{dom}(\sigma).(x \in S \rightarrow (\sigma'(x) = x))$ holds. We now generalize the above operation for a set of substitutions. Consider Σ' is a set of substitutions and $\Sigma = \Sigma' \setminus S$. We define $\Sigma = \Sigma' \setminus S$ in the following way, $\forall \sigma \in \Sigma'.(\Sigma \leftarrow \Sigma \cup \{\sigma \setminus S\})$. Given a substitution σ, we use $\sigma[x \mapsto t]$ to denote a new substitution which is same as σ except the variable x is now mapped to the new value t according to the new substitution. We now generalize the above operation for a set of substitutions. Consider Σ' is a set of substitutions and $\Sigma = \Sigma'[x \mapsto t]$. We define $\Sigma = \Sigma'[x \mapsto t]$ in the following way, $\forall \sigma \in \Sigma'.(\Sigma \leftarrow \Sigma \cup \{\sigma[x \mapsto t]\})$.

Given two substitutions σ_1 and σ_2 such that $\textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2) = \{\}$, we use $\sigma_1 + \sigma_2$ to denote the concatenation of the variable, value mappings of both σ_1 and σ_2. Consider $\sigma = \sigma_1 + \sigma_2$, then the following holds: $\forall x \in \textbf{dom}(\sigma).((x \in \textbf{dom}(\sigma_1) \rightarrow \sigma(x) = \sigma_1(x))) \wedge (x \in \textbf{dom}(\sigma_2) \rightarrow \sigma(x) = \sigma_2(x))))$. We also have: $\sigma + \bullet = \sigma$ and $\sigma + \sigma_\perp = \sigma_\perp$. If $\textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2) \neq \{\}$, then the substitution σ_1 for variables in $\textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2)$ is overridden by the substitution σ_2 for variables in $\textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2)$.

Given two substitutions σ_1 and σ_2, we use $\sigma_1 \bowtie \sigma_2$ to denote a new substitution which is the join of the two substitutions σ_1 and σ_2. Let $\sigma = \sigma_1 \bowtie \sigma_2$, σ is $\sigma_\perp$ when the following holds: $\exists x \in (\textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2)).(\sigma_1(x) \neq \sigma_2(x))$. When $\sigma \neq \{\}$ then the following holds: $\textbf{dom}(\sigma) = \textbf{dom}(\sigma_1) \cup \textbf{dom}(\sigma_2)$ and $\forall x \in \textbf{dom}(\sigma).(((x \in \textbf{dom}(\sigma_1) \wedge x \notin S) \rightarrow \sigma(x) = \sigma_1(x)) \wedge ((x \in \textbf{dom}(\sigma_2) \wedge x \notin S) \rightarrow \sigma(x) = \sigma_2(x)) \wedge (x \in S \rightarrow (\sigma(x) = \sigma_1(x) = \sigma_2(x))))$ where $S = \textbf{dom}(\sigma_1) \cap \textbf{dom}(\sigma_2)$. We consider the $\bowtie$ operation to be symmetric, that is $\sigma_1 \bowtie \sigma_2 = \sigma_2 \bowtie \sigma_1$. We also assume it is possible to calculate the join operation of two finite substitutions in some finite amount of time. We also have the following: $\sigma \bowtie \bullet = \sigma$ and $\sigma \bowtie \sigma_\perp = \sigma_\perp$. We use $\Join_{0 \leq k \leq j} \sigma_k$ to represent $\sigma_0 \bowtie \sigma_1 \bowtie \ldots \sigma_{j-1} \bowtie \sigma_j$. We write $\Join \sigma_k$ when the domain of k is understood from the context.

As the necessary notations have been introduced, we now discuss some obvious properties of substitutions.

Lemma 10 (Basic Substitution Properties)**.** *Let σ and σ' be arbitrary substitutions such that $\textbf{dom}(\sigma) \cap \textbf{dom}(\sigma') = \emptyset$, and let φ be any formula. Then*

1. *if $\textbf{dom}(\sigma) \cap fv(\varphi) = \emptyset$, then $\varphi\sigma = \varphi$,*
2. *$fv(\varphi\sigma) = fv(\varphi) \setminus \textbf{dom}(\sigma)$,*
3. *$\varphi(\sigma + \sigma') = (\varphi\sigma)\sigma' = (\varphi\sigma')\sigma$,*
4. *if $\textbf{dom}(\sigma) \supseteq fv(\varphi)$, then $fv(\varphi\sigma) = \emptyset$,*
5. *if $\textbf{dom}(\sigma) \supseteq fv(\varphi)$, then $\varphi\sigma = \varphi(\sigma + \sigma')$*
6. *$\varphi\sigma = \varphi(\sigma \downarrow fv(\varphi))$.*

Proof. The first three are by induction on the structure of φ. 4 follows from 2. 5 follows from 3, 4 and 1. 6 follows from 3 and 1. □

Lemma 11 (Substitution - fv Restriction and Extension)**.** *Let σ and σ' be substitutions and φ a formula such that $\textbf{dom}(\sigma) \cap \textbf{dom}(\sigma') = \emptyset$ and $\textbf{dom}(\sigma') \cap fv(\varphi) = \emptyset$. Then for all $\mathcal{L}$, j, η it holds that $\mathcal{L}, j, \eta \models \varphi(\sigma \downarrow fv(\varphi)) \iff \mathcal{L}, j, \eta \models \varphi\sigma \iff \mathcal{L}, j, \eta \models \varphi(\sigma + \sigma')$.*

Proof. By Lemma 10, we have $\varphi(\sigma \downarrow fv(\varphi)) = \varphi\sigma = \varphi(\sigma + \sigma')$. Thus the statement is trivially true. □

Corollary 1 (fv Substitution)**.** *Let σ be a substitution and φ a formula, with $\textbf{dom}(\sigma) \supseteq fv(\varphi)$. Then for all $\mathcal{L}$, j, η and $\sigma' \geq \sigma$ it holds that $\mathcal{L}, j, \eta \models \varphi\sigma \iff \mathcal{L}, j, \eta \models \varphi\sigma'$.*

Proof. Let $\sigma'' = \sigma \downarrow fv(\varphi)$. Then $\mathcal{L}, j, \eta \models \varphi\sigma'' \iff \mathcal{L}, j, \eta \models \varphi\sigma$ and $\mathcal{L}, j, \eta \models \varphi\sigma'' \iff \mathcal{L}, j, \eta \models \varphi\sigma'$ by previous lemma. Thus, $\mathcal{L}, j, \eta \models \varphi\sigma \iff \mathcal{L}, j, \eta \models \varphi\sigma'$. □

D.3 **précis** State (Π)

We now introduce the readers with the persistent *state* our algorithm **précis** uses to store the appropriate summary structures for each of the **B-formulas**. We denote the state with π.

A state π is a tuple of form $(\mathcal{A}, idx)$ where $\mathcal{A}$ has the type $Formula \rightarrow \mathbb{N} \rightarrow (\mathbb{S}_{\mathcal{S}} + \mathbb{S}_{\diamondminus} + \mathbb{S}_{\boxminus} + \mathbb{S}_{\ominus})$ whereas idx has the type $\mathbb{N}$. For a given π, we access the $\mathcal{A}$ component of it as $\pi.\mathcal{A}$ whereas we access the idx component of π using the notation $\pi.idx$. For any given $\pi = (\mathcal{A}, idx)$ we require that the $\mathcal{A}$ component of π be *well-formed* such that for a given $i \in \mathbb{N}$, $\mathcal{GMP}$ **B-formula** formulas of form: (1) $\alpha \, \mathcal{S}_{\mathbb{I}} \beta$ are mapped to $\mathbb{S}_{\mathcal{S}}$, (2) $\diamondminus_{\mathbb{I}}\varphi$ are mapped to $\mathbb{S}_{\diamondminus}$, (3) $\boxminus_{\mathbb{I}}\varphi$ are mapped to $\mathbb{S}_{\boxminus}$, and (4) $\ominus_{\mathbb{I}}\varphi$ are mapped to $\mathbb{S}_{\ominus}$.

We now describe the types of $\mathbb{S}_{\mathcal{S}}$, $\mathbb{S}_{\diamondminus}$, $\mathbb{S}_{\boxminus}$, and $\mathbb{S}_{\ominus}$. $\mathbb{S}_{\mathcal{S}}$ is a tuple $(\mathbb{S}_\alpha, \mathbb{S}_\beta, \mathbb{R})$ in which $\mathbb{S}_\alpha$ and $\mathbb{S}_\beta$ are structures containing pairs of form $\langle \sigma, k \rangle$ where σ is a substitution and $k \in \mathbb{N}$. $\mathbb{R}$ is a structure containing a set of substitutions. For a given state $\pi = (\mathcal{A}, idx)$ (where $\mathcal{A}$ is well-formed) and a $\mathcal{GMP}$ formula φ of form $\alpha \, \mathcal{S}_{\mathbb{I}} \beta$, we can access φ's structure $\mathbb{S}_\alpha$ at a specific position $i \in \mathbb{N}$ using the following notation: $\pi.\mathcal{A}(\alpha \, \mathcal{S}_{\mathbb{I}} \beta)(i).\mathbb{S}_\alpha$. For a given state $\pi = (\mathcal{A}, idx)$ (where $\mathcal{A}$ is well-formed) and a $\mathcal{GMP}$ formula φ of form $\alpha \, \mathcal{S}_{\mathbb{I}} \beta$, to specify that a specific $\langle \sigma, k \rangle$ is present in the structure $\mathbb{S}_\beta$ of φ at position $i \in \mathbb{N}$, we use the notation $\langle \sigma, k \rangle \in \pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta$. When the state ($\pi = (\mathcal{A}, idx)$) and the formula ($\varphi \equiv \alpha \, \mathcal{S}_{\mathbb{I}} \beta$) is understood from the context, we just write $\mathbb{S}^i_\alpha$ to express $\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha$. We follow these same notations for other structures too.

$\mathbb{S}_{\diamondminus}$ is a tuple $(\mathbb{P}, \mathbb{R})$ where $\mathbb{P}$ has the same type as $\mathbb{S}_\alpha$. In the same vein, $\mathbb{S}_{\boxminus}$ is a tuple $(\mathbb{H}, \mathbb{R})$ where $\mathbb{H}$ is a structure containing tuples of form $\langle \sigma, \mathit{left}, \mathit{right} \rangle$ where σ is a substitution and $\mathit{left}, \mathit{right} \in \mathbb{N}$. $\mathbb{S}_{\ominus}$ is a tuple $(\mathbb{T}, \mathbb{R})$ where $\mathbb{T}$ has the same type as $\mathbb{R}$.

Next we introduce the readers with what we mean by a *well-formed state* and define two properties (*weak consistency* and *strong consistency*) of a well-formed state which we use in the correctness lemma later. However, first we define what we mean by *buildable strict temporal subformulas* of a given formula φ. We use the concept of buildable strict temporal subformulas of a given formula φ while defining weak and strong consistency of a well-formed state.

Definition 5 (Buildable Strict Temporal Sub-formula)**.** *Given a formula φ, the set of buildable strict temporal sub-formulas of φ is denoted by* $\textsf{b-s-tsub}(\varphi)$ *and is defined as* $\textsf{b-s-tsub}(\varphi) = \textsf{b-tsub}(\varphi) \setminus \{\varphi\}$.

We now define what it means for a state π to be well-formed at a specific position $j \in \mathbb{N}$ with respect to a log $\mathcal{L}$ and a formula φ where $\mathbf{B} \in \mathit{label}(\varphi)$.

Definition 6 (Well-formed State, Ψ)**.** *Given a state $\pi = (\mathcal{A}, idx)$ (where $idx \in \mathbb{N}$ and $\pi.\mathcal{A}$ is well-formed), we say π is well-formed at $j \in \mathbb{N}$ (where $j \leq idx$) with respect to a log $\mathcal{L}$, time stamp sequence τ, and a formula φ of form $\alpha \, \mathcal{S}_{\mathbb{I}} \beta$, $\diamondminus_{\mathbb{I}}\alpha$, $\boxminus_{\mathbb{I}}\alpha$, or $\ominus_{\mathbb{I}}\alpha$ where $\emptyset \vdash_{\boldsymbol{B}} \varphi : \chi_O$ such that $\chi_O \subseteq fv(\varphi)$, denoted by $\Psi(\mathcal{L}, \tau, \pi, \varphi, j)$, if all of the following hold:*

$$\textsf{b-tsub}(\varphi) = \begin{cases} \emptyset & \text{if } \varphi \equiv \top|\bot|\mathsf{p}(t_1,\ldots,t_n) \\ \textsf{b-tsub}(\varphi_1) \cup \textsf{b-tsub}(\varphi_2) & \text{if } \varphi \equiv \varphi_1 \vee \varphi_2|\varphi_1 \wedge \varphi_2|\varphi_1\,\mathcal{U}_{\mathbb{I}}\varphi_2 \\ \{\varphi\} \cup \textsf{b-tsub}(\alpha) \cup \textsf{b-tsub}(\beta) & \text{if } \varphi \equiv \alpha\,\mathcal{S}_{\mathbb{I}}\beta \text{ and } \mathbf{B} \in \mathit{label}(\varphi) \\ \textsf{b-tsub}(\alpha) \cup \textsf{b-tsub}(\beta) & \text{if } \varphi \equiv \alpha\,\mathcal{S}_{\mathbb{I}}\beta \text{ and } \mathbf{B} \notin \mathit{label}(\varphi) \\ \textsf{b-tsub}(\varphi_1) \cup \textsf{b-tsub}(\varphi_2) & \text{if } \varphi \equiv \forall\vec{x}.(\varphi_1 \rightarrow \varphi_2) \\ \textsf{b-tsub}(\varphi) & \text{if } \varphi \equiv \exists\vec{x}.\varphi \\ \textsf{b-tsub}(\varphi) \cup \{\varphi\} & \text{if } \varphi \equiv \diamondminus_{\mathbb{I}}\varphi \mid \boxminus_{\mathbb{I}}\varphi \mid \ominus_{\mathbb{I}}\varphi \text{ and } \mathbf{B} \in \mathit{label}(\varphi) \\ \textsf{b-tsub}(\varphi) & \text{if } \varphi \equiv \diamondminus_{\mathbb{I}}\varphi \mid \boxminus_{\mathbb{I}}\varphi \mid \ominus_{\mathbb{I}}\varphi \text{ and } \mathbf{B} \notin \mathit{label}(\varphi) \\ \textsf{b-tsub}(\varphi) & \text{if } \varphi \equiv \Diamond_{\mathbb{I}}\varphi \mid \Box_{\mathbb{I}}\varphi \mid \bigcirc_{\mathbb{I}}\varphi \end{cases}$$

Figure 12: Function definition: $\textsf{b-tsub}(\varphi)$

- $\varphi \equiv \alpha\,\mathcal{S}_{[c,d]}\beta$, $\emptyset \vdash_{\boldsymbol{B}} \beta : \chi_1$, *and* $\chi_1 \vdash_{\boldsymbol{B}} \alpha : \chi_2$ *implies that:*

  1. *(**SOUNDNESS**-$\pi.\mathcal{A}(j)(\varphi).\mathbb{S}_\alpha$)*
     $\forall\langle\sigma,k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\alpha, \eta.\textsf{dom}(\sigma) \supseteq (\chi_1 \cup \chi_2) \wedge (\forall\sigma', l.(k \leq l \leq j) \wedge \sigma' \geq \sigma \rightarrow \mathcal{L},\tau,l,\eta \models \alpha\sigma')$.
  2. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\alpha$)*
     $\forall\sigma,k,\sigma_\beta,\eta.(k < j) \wedge \langle\sigma_\beta,k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\beta \wedge \sigma \geq \sigma_\beta \wedge \textsf{dom}(\sigma) \supseteq (\chi_1 \cup \chi_2) \wedge \forall l.(k+1 \leq l \leq j) \wedge \mathcal{L},\tau,l,\eta \models \alpha\sigma \rightarrow \exists\sigma', m.\sigma \geq \sigma' \wedge m \leq (k+1) \wedge \langle\sigma',m\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\alpha$.
  3. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\beta$)*
     $\forall\langle\sigma,k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\beta, \eta.\textsf{dom}(\sigma) \supseteq \chi_1 \wedge \tau_j - \tau_k \leq d \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,k,\eta \models \beta\sigma')$.
  4. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\beta$)*
     $\forall\sigma,k,\eta.\textsf{dom}(\sigma) \supseteq fv(\beta) \wedge \tau_j - \tau_k \leq d \wedge \mathcal{L},\tau,k,\eta \models \beta\sigma \rightarrow (\exists\langle\sigma',k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{S}_\beta.\ \sigma \geq \sigma')$.
  5. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
     $\forall\sigma \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}, \eta.\textsf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,j,\eta \models \varphi\sigma')$.
  6. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
     $\forall\sigma,\eta.\textsf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L},\tau,j,\eta \models \varphi\sigma \rightarrow (\exists\sigma' \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}.\ \sigma \geq \sigma')$.

- $\varphi \equiv \ominus_{[c,d]}\alpha$ *implies that:*

  1. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{T}$)*
     $\forall\sigma \in \pi.\mathcal{A}(\varphi)(j).\mathbb{T}, \eta.\textsf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,j,\eta \models \alpha\sigma')$.
  2. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{T}$)*
     $\forall\sigma,\eta.\textsf{dom}(\sigma) \supseteq fv(\alpha) \wedge \mathcal{L},\tau,j,\eta \models \alpha\sigma \rightarrow (\exists\sigma' \in \pi.\mathcal{A}(\varphi)(j).\mathbb{T}.\sigma \geq \sigma')$.
  3. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
     $\forall\sigma \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}, \eta.\textsf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,j,\eta \models \varphi\sigma')$.
  4. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
     $\forall\sigma,\eta.\textsf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L},\tau,j,\eta \models \varphi\sigma \rightarrow (\exists\sigma' \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}.\sigma \geq \sigma')$.

- $\varphi \equiv \diamondminus_{[c,d]}\alpha$ *implies that:*

1. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{P}$)*
   $\forall\langle\sigma,k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{P}, \eta.\textbf{\textsf{dom}}(\sigma) \supseteq \chi_O \wedge \tau_j - \tau_k \leq d \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,k,\eta \models \alpha\sigma')$.
2. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{P}$)*
   $\forall\sigma,k,\eta.\textbf{\textsf{dom}}(\sigma) \supseteq fv(\alpha) \wedge \tau_j - \tau_k \leq d \wedge \mathcal{L},\tau,k,\eta \models \alpha\sigma \rightarrow (\exists\langle\sigma',k\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{P}.\sigma \geq \sigma')$.
3. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
   $\forall\sigma \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}, \eta.\textbf{\textsf{dom}}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,j,\eta \models \varphi\sigma')$.
4. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
   $\forall\sigma,\eta.\textbf{\textsf{dom}}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L},\tau,j,\eta \models \varphi\sigma \rightarrow (\exists\sigma' \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}.\sigma \geq \sigma')$.

- $\varphi \equiv \boxminus_{[c,d]}\alpha$ *implies that:*

1. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{H}$)*
   $\forall\langle\sigma,left,right\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{H}, \eta.\textbf{\textsf{dom}}(\sigma) \supseteq \chi_O \wedge \tau_j - \tau_{right} \leq d \wedge \forall\sigma', l \in [left,right].(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,l,\eta \models \alpha\sigma')$.
2. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{H}$)*
   $\forall\sigma,L,R,\eta.\textbf{\textsf{dom}}(\sigma) \supseteq fv(\alpha) \wedge (L \leq R \leq j) \wedge (\tau_j - \tau_R \leq d) \wedge (\forall t.(L \leq t \leq R) \rightarrow \mathcal{L},\tau,t,\eta \models \alpha\sigma) \rightarrow \exists\sigma', left, right.(\sigma \geq \sigma') \wedge left \leq L \leq R \leq right \leq j \wedge \langle\sigma',left,right\rangle \in \pi.\mathcal{A}(\varphi)(j).\mathbb{H}$.
3. *(**SOUNDNESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
   $\forall\sigma \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}, \eta.\textbf{\textsf{dom}}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L},\tau,j,\eta \models \varphi\sigma')$.
4. *(**COMPLETENESS**-$\pi.\mathcal{A}(\varphi)(j).\mathbb{R}$)*
   $\forall\sigma,\eta.\textbf{\textsf{dom}}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L},\tau,j,\eta \models \varphi\sigma \rightarrow (\exists\sigma' \in \pi.\mathcal{A}(\varphi)(j).\mathbb{R}.\sigma \geq \sigma')$.

As we have already introduced what it means for a state to be well-formed, we now formally define two properties of a well-formed state, which we use in the statement of the subsequent correctness lemmas, based on the definition of a well-formed state Ψ. Moreover, from now on, when we mention a state we mean a well-formed state unless explicitly mentioned otherwise.

Definition 7 (Strong Consistency). *A state $\pi = (\mathcal{A}, idx)$ (where $idx \in \mathbb{N}$) is* strongly consistent *at $j \in \mathbb{Z}$ (where $j \leq idx$) with respect to a log $\mathcal{L}$, a time stamp sequence τ, and a formula φ if: (1) $j < 0$ or (2) for all $\hat{\varphi} \in \textsf{b-tsub}(\varphi)$ and for all $0 \leq k \leq j$, $\Psi(\mathcal{L},\tau,\pi,\hat{\varphi},k)$ holds.*

Definition 8 (Weak Consistency). *A state $\pi = (\mathcal{A}, idx)$ (where $idx \in \mathbb{N}$) is* weakly consistent *at $j \in \mathbb{Z}$ (where $j \leq idx$) with respect to a log $\mathcal{L}$, a time stamp sequence τ, and a formula φ if: (1) $j < 0$ or (2) π is a strongly consistent state at $j-1$ with respect to $\mathcal{L}$ and φ, and additionally for all $\hat{\varphi} \in \textsf{b-s-tsub}(\varphi)$, $\Psi(\mathcal{L},\tau,\pi,\hat{\varphi},j)$ holds.*

We now define the size of a state with respect to a temporal **`B-formula`** φ and a position $j \in \mathbb{Z}$. The size of a state at a position j with respect to a **`B-formula`** φ, a log $\mathcal{L}$, and a time stamp sequence τ, is the summation of summary structure size of φ at all position k where $0 \leq k \leq j$. The finiteness of the state with respect to all **`B-formula`** of a given policy φ we enable us to show the termination of our algorithm.

Definition 9 (Size of a state with respect to a buildable temporal formula). *Given a log $\mathcal{L}$, a time stamp sequence τ, a formula φ of form $\alpha\,\mathcal{S}_{\mathbb{I}}\beta$, $\diamondminus_{\mathbb{I}}\alpha$, $\boxminus_{\mathbb{I}}\alpha$, or $\ominus_{\mathbb{I}}\alpha$ such that $\emptyset \vdash_{\boldsymbol{B}} \varphi : \chi_O$ where $\chi_O \subseteq fv(\varphi)$, a state $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, a position $j \in \mathbb{Z}$ such that $j \leq i$ and π is strongly* *consistent at j with respect to $\mathcal{L}$, τ, and φ, then the size of π at j with respect to φ, denoted by $\Upsilon(\pi, j, \varphi)$, is defined as follows:*

$$\Upsilon(\pi, j, \varphi) = \begin{cases} 0 & \textit{if } j < 0 \\ \sum\limits_{0 \le k \le j} (|\pi.\mathcal{A}(\varphi)(k).\mathbb{S}_\alpha| + |\pi.\mathcal{A}(\varphi)(k).\mathbb{S}_\beta| + |\pi.\mathcal{A}(\varphi)(k).\mathbb{R}|) & \textit{if } \varphi \equiv \alpha\, \mathcal{S}_{\mathbb{I}} \beta \\ \sum\limits_{0 \le k \le j} (|\pi.\mathcal{A}(\varphi)(k).\mathbb{P}| + |\pi.\mathcal{A}(\varphi)(k).\mathbb{R}|) & \textit{if } \varphi \equiv \diamondminus_{\mathbb{I}} \alpha \\ \sum\limits_{0 \le k \le j} (|\pi.\mathcal{A}(\varphi)(k).\mathbb{H}| + |\pi.\mathcal{A}(\varphi)(k).\mathbb{R}|) & \textit{if } \varphi \equiv \boxminus_{\mathbb{I}} \alpha \\ \sum\limits_{0 \le k \le j} (|\pi.\mathcal{A}(\varphi)(k).\mathbb{T}| + |\pi.\mathcal{A}(\varphi)(k).\mathbb{R}|) & \textit{if } \varphi \equiv \ominus_{\mathbb{I}} \alpha \end{cases}$$

D.4 Correctness

Now that we have established all the necessary notions to understand the correctness of our algorithm **précis**, we first present several lemmas describing the properties of the different functions our algorithm **précis**. This will be necessary to show the termination and correctness of **précis**.

We start of by describing the property of the **sat** function. Note that we use the **sat** function as the building block of the **ips** function. The **sat** function satisfies the following claim.

Claim 1 (**sat** function)**.** *Given a log $\mathcal{L}$, a position $j \in \mathbb{N}$, a time stamp sequence τ, an input substitution σ_{in}, and a predicate $\mathsf{p}(\vec{t})$ such that for all $k \in I(\mathsf{p})$ the following holds: $\forall x \in fv(t_k).x \in$ **dom**(σ_{in}), then **sat**$(\mathcal{L}, j, \tau, \mathsf{p}(\vec{t}), \sigma_{\text{in}})$* terminates *and returns the* finite *set of all substitutions Σ_{out} for variables in $\bigcup_{k \in I(\mathsf{p})} fv(t_k) \cup \bigcup_{i \in O(\mathsf{p})} fv(t_i) \cup$ **dom**(σ_{in}), where for all $\sigma \in \Sigma_{out}$ and η, $\sigma \ge \sigma_{\text{in}}$ and $\mathcal{L}, \tau, j, \eta \models \mathsf{p}(\vec{t})\sigma$ hold.*

We now present a lemma about the **ips** function which states that all the substitutions returned by the **ips** function is actually an extension of the input substitution σ_{in} that the **ips** function takes as an argument. It can be shown that the semantics of $\mathcal{GMP}$ formulas and properties of it are invariant under renaming quantified variables.

Lemma 12 (**ips** is Extension)**.** *For all formulas φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all states $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for all substitutions σ_{in}, for any χ_C and χ_F, such that: (1) $\chi_C, \chi_F \vdash \varphi : \chi_O$, (2) $i \ge j$ and $\tau_i - \tau_j \ge \Delta(\varphi)$, (3) **dom**$(\sigma_{in}) \supseteq \chi_C \cup \chi_F$, (4) π is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$, if **ips**$(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi) = \Sigma_{out}$, then for all $\sigma \in \Sigma_{out}$ it holds that $\sigma \ge \sigma_{in}$.*

Proof. The proof proceeds by doing an induction on the structure of φ. We show select cases and the other cases are similar.

Case $\varphi \equiv \top$.
Then $\Sigma_{out} = \{\sigma_{in}\}$, so $\sigma_{in} \le \sigma_{in}$.

Case $\varphi \equiv \bot$.
Since $\Sigma_{out} = \emptyset$, the statement is vacuously true.

Case $\varphi \equiv \mathsf{p}(t_1, \ldots, t_n)$.
Then $\Sigma_{out} =$ **sat**$(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}})$, and by Claim 1 we have, $\forall \sigma \in$ **sat**$(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}}).\sigma \ge \sigma_{in}$.

Case $\varphi \equiv \varphi_1 \vee \varphi_2$.

Then $\Sigma_{out} = \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1) \cup \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$. W.l.o.g., $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$. Inspection of the applicable mode checking judgements verifies that the inductive hypothesis is applicable, which yields that $\sigma \geq \sigma_{in}$.

Case $\varphi \equiv \varphi_1 \wedge \varphi_2$.

Then $\Sigma_{out} = \bigcup_{\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_1)} \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. If $\sigma \in \Sigma_{out}$, then there exists $\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_1)$ such that $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. Inspection of the applicable mode checking judgements verifies that the inductive hypothesis is applicable, which first yields that $\sigma_c \geq \sigma_{in}$, and then $\sigma \geq \sigma_c$. By transitivity of $\geq$, we have $\sigma \geq \sigma_{in}$.

Case $\varphi \equiv \exists \vec{x}.\varphi$.

W.l.o.g., we have $\texttt{dom}(\sigma_{in}) \cap \vec{x} = \emptyset$ as we can rename $\vec{x}$ to some fresh $\vec{y}$. Then $\Sigma_{out} = \texttt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \varphi) \setminus \{\vec{x}\}$. Then there exists $\sigma' \in \texttt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \varphi)$ such that $\sigma = \sigma' \setminus \{\vec{x}\}$. By inductive hypothesis, $\sigma' \geq \sigma_{in}$ from which we have $\sigma' \setminus \{\vec{x}\} \geq \sigma_{in} \setminus \{\vec{x}\}$. As $\sigma = \sigma' \setminus \{\vec{x}\}$, we now have $\sigma \geq \sigma_{in} \setminus \{\vec{x}\}$ and as $\texttt{dom}(\sigma_{in}) \cap \vec{x} = \emptyset$, we have $\sigma_{in} \setminus \{\vec{x}\} = \sigma_{in}$. Then $\sigma \geq \sigma_{in}$.

Case $\varphi \equiv \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$.

If $\Sigma_{out} = \{\}$, then the statement vacuously holds. Else $\Sigma_{out} = \{\sigma_{in}\}$, so $\sigma_{in} \leq \sigma_{in}$.

Case $\varphi \equiv \varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2$.

Sub-Case $\mathbf{B} \in \mathit{label}(\varphi)$.

Then $\Sigma_{out} = \sigma_{in} \bowtie \pi.\mathcal{A}(\varphi_1 \, \mathcal{S} \, \varphi_2)(i).\mathbb{R}$, so by $\bowtie$ properties $\forall \sigma \in \Sigma_{out}.\sigma \geq \sigma_{in}$.

Sub-Case $\mathbf{B} \notin \mathit{label}(\varphi)$.

Then $\sigma \in S_{R_1}$ or $\sigma \in S_{R_2}$.

Sub-Sub-Case $\sigma \in S_{R_1}$.

Then $\langle \sigma, j \rangle \in S_{\varphi_2}$, so $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, $\sigma \geq \sigma_{in}$.

Sub-Sub-Case $\sigma \in S_{R_2}$.

Then $\sigma \in S_{\varphi_1} = \{\Join \sigma'_l \neq \sigma_\perp | \exists \langle \sigma_\beta, k \rangle \in S_{\varphi_2}.k < j \wedge \forall l.(k < l \leq j \rightarrow \sigma'_l \in \texttt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \varphi_1))\}$. Then $\sigma = \Join \sigma'_l$ for some σ'_l with a certain k, so $\forall l.k < l \leq j \rightarrow \sigma \geq \sigma'_l$ by $\bowtie$ properties. By inductive hypothesis, $\forall l.k < l \leq j \rightarrow \sigma'_l \geq \sigma_\beta$, thus $\sigma \geq \sigma_\beta$. Since $\langle \sigma_\beta, k \rangle \in S_{\varphi_2}$, $\sigma_\beta \in \texttt{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, $\sigma_\beta \geq \sigma_{in}$. By transitivity, $\sigma \geq \sigma_{in}$.

Case $\varphi \equiv \varphi_1 \, \mathcal{U}_{[c,d]} \varphi_2$.

Then $\sigma \in S_{R_1}$ or $\sigma \in S_{R_2}$.

Sub-Case $\sigma \in S_{R_1}$.

Then $\langle \sigma, j \rangle \in S_{\varphi_2}$, so $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, $\sigma \geq \sigma_{in}$.

Sub-Case $\sigma \in S_{R_2}$.

Then $\sigma \in S_{\varphi_1} = \{\Join \sigma^\alpha_l \neq \sigma_\perp | \exists \langle \sigma_\beta, k \rangle \in S_{\varphi_2}.k \neq i \wedge \forall (i \leq l < k).\sigma^\alpha_l \in \texttt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \alpha)\}$. Then $\sigma = \Join \sigma'_l$ for some σ'_l with a certain k, so $\forall l.j \leq l < k \rightarrow \sigma \geq \sigma'_l$ by $\bowtie$ properties. By inductive hypothesis, $\forall l.j \leq l < k \rightarrow \sigma'_l \geq \sigma_\beta$, thus $\sigma \geq \sigma_\beta$. Since $\langle \sigma_\beta, k \rangle \in S_{\varphi_2}$, $\sigma_\beta \in \texttt{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, $\sigma_\beta \geq \sigma_{in}$. By transitivity, $\sigma \geq \sigma_{in}$.

□

Lemma 13 (Upper Bound of Substitutions returned by **ips** and Stored in the State)**.** *1. For all $\mathcal{GMP}$ formulas φ either of form $\varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$, $\diamondminus_{\mathbb{I}}\varphi$, $\boxminus_{\mathbb{I}}\varphi$, or $\ominus_{\mathbb{I}}\varphi$, such that $\{\} \vdash_{\boldsymbol{B}} \varphi : \chi_O$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for a specific $i \in \mathbb{N}$, for all states $\pi = (\mathcal{A}, i)$ where π is strongly consistent at i with respect to $\mathcal{L}$, τ, and φ, $\forall\sigma(\sigma \in \pi.\mathcal{A}(\varphi)(i).\mathbb{R} \rightarrow \textbf{dom}(\sigma) \subseteq fv(\varphi))$.*

2. For all $\mathcal{GMP}$ formulas φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all empty environments η_0 for all state $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for all substitutions σ_{in}, for some given χ_C and χ_F, such that: (1) $\chi_C, \chi_F \vdash \varphi : \chi_O$, (2) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, (3) $\textbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$, (4) π is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$, if $\textbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi) = \Sigma_{out}$ then $\forall\sigma(\sigma \in \Sigma_{out} \rightarrow \textbf{dom}(\sigma) \subseteq \textbf{dom}(\sigma_{in}) \cup fv(\varphi))$.

Proof. The proof is straightforward by structural induction on the policy φ and by using the construction of **ips** and **uSS**. □

Lemma 14 (Correctness of **uSS** and **ips** function)**.** *1. For all $\mathcal{GMP}$ formulas φ either of form $\varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$, $\diamondminus_{\mathbb{I}}\varphi$, $\boxminus_{\mathbb{I}}\varphi$, or $\ominus_{\mathbb{I}}\varphi$, such that $\{\} \vdash_{\boldsymbol{B}} \varphi : \chi_O$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for a specific $i \in \mathbb{N}$, for all states $\pi = (\mathcal{A}, i)$ where π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ, if $\textbf{uSS}(\mathcal{L}, i, \tau, \pi, \varphi) = \hat{\pi}$ then $\hat{\pi} = (\mathcal{A}', i)$ is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$.*

2. For all $\mathcal{GMP}$ formulas φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all empty environments η_0 for all state $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for all substitutions σ_{in}, for some given χ_C and χ_F, such that: (1) $\chi_C, \chi_F \vdash \varphi : \chi_O$, (2) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, (3) $\textbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$, (4) π is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$, if $\textbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi) = \Sigma_{out}$ then the following holds:

*(a) **(SOUNDNESS)***
$\forall\sigma \in \Sigma_{out}.(\textbf{dom}(\sigma) \supseteq (\chi_O \cup \chi_C \cup \chi_F) \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta_0 \models \varphi\sigma'))$.

*(b) **(COMPLETENESS)***
$\forall\sigma.((\sigma \geq \sigma_{in} \wedge \textbf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, j, \eta_0 \models \varphi.\sigma) \rightarrow (\exists\sigma_o \in \Sigma_{out}.(\sigma \geq \sigma_o)))$.

Proof. We first show the proof of part (1) then part (2). Note that, when **uSS** is called for a **B-formula** φ, it calls **ips** on strict subformulas of φ. However, **ips** does not call **uSS** directly and hence we do not have any cyclic dependency. The proof does a mutual induction on the structure of the policy φ.

Proof of part (1): Mutual induction on the structure of φ.

We are given that the state $\pi = (\mathcal{A}, i)$ is weakly consistent at trace position i with respect to $\mathcal{L}$, τ, and φ. We then have to show that the state $\hat{\pi} = (\mathcal{A}', i)$ is strongly consistent at trace position i with respect to $\mathcal{L}$, τ, and φ. From Definition 7 and 8, it is thus sufficient to show that $\hat{\pi}$ is well-formed at i with respect to $\mathcal{L}$, τ, and φ. In other words, it is sufficient to show that $\Psi(\mathcal{L}, \tau, \hat{\pi}, \varphi, i)$ holds.

Case $\varphi \equiv \ominus_{[c,d]}\alpha$

(Soundness)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}$

We have to show that $\forall\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma')$. Take any arbitrary $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}$. From construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}$ we know that $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T} \leftarrow \mathtt{ips}(\mathcal{L}, i, \tau,$
$\pi, \bullet, \alpha)$. Hence, $\sigma \in \mathtt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha)$. From the applicable mode checking judgement, we have $\{\} \vdash_{\mathbf{B}} \ominus_{[c,d]}\alpha : \chi_O$ where $\chi_O \subseteq fv(\alpha)$. It follows that $\{\} \vdash_{\mathbf{B}} \alpha : \chi_O$. By Lemma 7, we have $\{\}, \{\} \vdash \alpha : \chi_O$. By Lemma 4, we have $\Delta(\ominus_{[c,d]}\alpha) = 0$. Moreover, $\mathbf{dom}(\bullet) \supseteq \chi_C \cup \chi_F$ where $\chi_C = \chi_F = \{\}$. As π is weakly consistent at position i with respect to $\mathcal{L}$, τ, and $\ominus_{[c,d]}\alpha$, we know that π is strongly consistent at position i with respect to $\mathcal{L}$, τ, and α. We see that we have satisfied all the premises of the soundness of $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha)$. By i.h., we have $\forall\sigma \in \Sigma_{out}.(\mathbf{dom}(\sigma) \supseteq (\chi_O \cup \chi_C \cup \chi_F) \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta_0 \models \varphi\sigma'))$ where $\Sigma_{out} \leftarrow \mathtt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha)$. We have $\chi_C = \chi_F = \{\}$, $\varphi = \alpha$, and $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T} = \Sigma_{out}$. Hence, we have $\forall\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma')$ where $\eta = \eta_0$.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We can have the following two cases.

Sub-Sub-Case $i = 0$

By construction $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R} = \emptyset$ hence the statement is vacuously true.

Sub-Sub-Case $i > 0$

We have to show that $\forall\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$. Take any arbitrary $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$. By construction $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R} = \{\sigma \mid \sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{T} \wedge (c \leq \tau_i - \tau_{(i-1)} \leq d)\}$. As π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ, from the definition, we can say that it is strongly consistent at $(i-1)$ with respect to $\mathcal{L}$, τ, and α. From the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{T}$, we have $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{T}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i-1, \eta \models \alpha\sigma')$. From the semantics of $\ominus$ we have, $\mathcal{L}, \tau, i, \eta \models \ominus_{[c,d]}\alpha\sigma \iff i > 0 \wedge \mathcal{L}, \tau, i-1, \eta \models \alpha\sigma \wedge c \leq \tau_i - \tau_{(i-1)} \leq d$. By construction we have $(c \leq \tau_i - \tau_{(i-1)} \leq d)$ and from $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{T}$ soundness and semantics of $\ominus$, we have $\forall\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$.

(**Completeness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}$

We have to show that $\forall\sigma, \eta.\mathbf{dom}(\sigma) \supseteq fv(\alpha) \wedge \mathcal{L}, \tau, i, \eta \models \alpha\sigma \rightarrow (\exists\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}.\sigma \geq \sigma')$. With the same arguments as in soundness, we can show that we satisfy the premise required to use the completeness statement for $\mathtt{ips}$. From completeness of $\mathtt{ips}$, we have: $\forall\sigma.((\sigma \geq \sigma_{in} \wedge \mathbf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, i, \eta_0 \models \varphi.\sigma) \rightarrow (\exists\sigma_o \in \Sigma_{out}.(\sigma \geq \sigma_o)))$ where $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T} = \Sigma_{out}$. Having $\eta = \eta_0$, $\sigma_{in} = \bullet$, $\varphi = \alpha$ we have, $\forall\sigma, \eta.\mathbf{dom}(\sigma) \supseteq fv(\alpha) \wedge \mathcal{L}, \tau, i, \eta \models \alpha\sigma \rightarrow (\exists\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{T}.\sigma \geq \sigma')$.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We can have the following two cases.

Sub-Sub-Case $i = 0$

By semantics of $\ominus$, there is no σ for which $\mathcal{L}, \tau, 0, \eta_0 \models \ominus_{[c,d]}\alpha$ holds hence the statement is vacuously true.

Sub-Sub-Case $i > 0$

We have to show that $\forall \sigma, \eta . \mathbf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, i, \eta \models \varphi\sigma \rightarrow (\exists \sigma' \in \pi.\mathcal{A}(\varphi)(i).\mathbb{R}.\sigma \geq \sigma')$. Take any arbitrary σ such that $\mathbf{dom}(\sigma) \supseteq fv(\varphi)$ and $\mathcal{L}, \tau, i, \eta \models \varphi\sigma$. From the semantics of $\ominus$ we know, $\mathcal{L}, \tau, i, \eta \models \ominus_{[c,d]}\alpha\sigma \iff i > 0 \wedge \mathcal{L}, \tau, i-1, \eta \models \alpha\sigma \wedge c \leq \tau_i - \tau_{(i-1)} \leq d$. As π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ, from the definition, we can say that it is strongly consistent at $(i-1)$ with respect to $\mathcal{L}$, τ, and α. From completeness of $\pi.\mathcal{A}(\varphi)(i-1).\mathbb{T}$, we know that there exists $\sigma_o \in \pi.\mathcal{A}(\varphi)(i-1).\mathbb{T}$ such that $\sigma \geq \sigma_o$. From $c \leq \tau_i - \tau_{(i-1)} \leq d$, the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, and the above we have $\sigma_o \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$.

Case $\varphi \equiv \diamondminus_{[c,d]}\alpha$

(**Soundness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$

We have to show that $\forall \langle \sigma, k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}, \eta . \mathbf{dom}(\sigma) \supseteq \chi_O \wedge \tau_i - \tau_k \leq d \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, k, \eta \models \alpha\sigma')$. Take any arbitrary $\langle \sigma, k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$. By construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, we have $\langle \sigma, k \rangle \in S_a$ or $\langle \sigma, k \rangle \in (\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P} \setminus S_r)$.

Sub-Sub-Case $\langle \sigma, k \rangle \in S_a$

By construction, $S_a \leftarrow \{\langle \sigma, i \rangle \mid \sigma \in \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha) \wedge 0 \leq d\}$ and $k = i$. By checking the applicable mode checking judgements, by Lemma 7, by Lemma 4, and from the premise, we see that the premise of $\mathbf{ips}$ soundness is satisfied. By i.h. of $\mathbf{ips}$ soundness, we have $(\mathbf{dom}(\sigma) \supseteq (\chi_O \cup \chi_C \cup \chi_F) \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta_0 \models \alpha\sigma'))$. As $\chi_C = \chi_F = \{\}$, $\eta = \eta_0$, and from construction we have $\tau_i - \tau_k \leq d$, hence we have $\mathbf{dom}(\sigma) \supseteq (\chi_O)$ and $\forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, k, \eta \models \alpha\sigma')$, concluding our proof.

Sub-Sub-Case $\langle \sigma, k \rangle \in (\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P} \setminus S_r)$

By construction, $\langle \sigma, k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P}$ and $\tau_i - \tau_k \leq d$. As we know that π is strongly consistent at $(i-1)$ with respect to $\mathcal{L}$, τ, and π, from soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P}$, we have $\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \tau_{(i-1)} - \tau_k \leq d \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, k, \eta \models \alpha\sigma')$. From construction we additionally have $\tau_i - \tau_k \leq d$. Hence we conclude $\forall \langle \sigma, k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}, \eta . \mathbf{dom}(\sigma) \supseteq \chi_O \wedge \tau_i - \tau_k \leq d \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, k, \eta \models \alpha\sigma')$.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall \sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}, \eta . \mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$. Take any arbitrary $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$. From construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$ we have $\exists k . \langle \sigma, k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$ and also $(\tau_i - \tau_k \in [c, d])$. Hence from the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$ we have $\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \tau_i - \tau_k \leq d \wedge \forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, k, \eta \models \alpha\sigma')$. From the semantics of $\diamondminus_{[c,d]}$, we have $\mathcal{L}, \tau, i, \eta \models \diamondminus_{[c,d]}\alpha \iff \exists k \in \mathbb{N} . k \leq i \wedge \mathcal{L}, \tau, k, \eta \models \alpha \wedge (\tau_i - \tau_k \in [c, d])$. Thus according to semantics, from the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$ and by $(\tau_i - \tau_k \in [c, d])$ (from construction), we have $\mathbf{dom}(\sigma) \supseteq \chi_O$ and $\forall \sigma' . (\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$, concluding our proof.

(**Completeness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$

We have to show that $\forall \sigma, k, \eta . \mathbf{dom}(\sigma) \supseteq fv(\alpha) \wedge \tau_i - \tau_k \leq d \wedge \mathcal{L}, \tau, k, \eta \models \alpha\sigma \rightarrow (\exists \langle \sigma', k \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}.\sigma \geq \sigma')$. Take any arbitrary σ and k such that $\mathbf{dom}(\sigma) \supseteq fv(\alpha)$, $\tau_i - \tau_k \leq d$, and $\mathcal{L}, \tau, k, \eta \models \alpha\sigma$. We can have the following two cases:

Sub-Sub-Case $k = i$

In the same vein of the soundness proof of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, we see that the premise of **ips** completeness is satisfied. From **ips** completeness we see that, $\exists\sigma_o \in \textbf{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha).(\sigma \geq \sigma_o)$. From this, $\tau_i - \tau_i = 0 \leq d$, and $\sigma' = \sigma_o$, according to the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, $\sigma' \in S_a$ and in turn $\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, completing our proof.

Sub-Sub-Case $k < i$

Note that we are given that π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ. From the definition, we know that π is strongly consistent at $(i-1)$ with respect to $\mathcal{L}$, τ, and α. From premise we know that $\mathcal{L}, \tau, k, \eta \models \alpha\sigma$ where $k < i$. From the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P}$, we have the following: $\exists\langle\sigma', k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P}.\sigma \geq \sigma'$. From the above and $\tau_i - \tau_k \leq d$, according to the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, we know that $\sigma' \in (\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{P} \setminus S_r)$ and hence $\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, completing our proof.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall\sigma, \eta.\textbf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, i, \eta \models \varphi\sigma \rightarrow (\exists\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}.\sigma \geq \sigma')$. Take any arbitrary σ such that $\textbf{dom}(\sigma) \supseteq fv(\varphi)$ and $\mathcal{L}, \tau, i, \eta \models \diamondminus_{[c,d]}\alpha\sigma$. From the semantics of $\diamondminus$ we have, $\mathcal{L}, \tau, i, \eta \models \diamondminus_{[c,d]}\alpha\sigma \iff \exists k.k \leq i \wedge c \leq \tau_i - \tau_k \leq d \wedge \mathcal{L}, \tau, k, \eta \models \alpha\sigma$. As $fv(\varphi) = fv(\alpha)$, $\tau_i - \tau_k \leq d$, and $\mathcal{L}, \tau, k, \eta \models \alpha\sigma$, from the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$, we have $\exists\langle\sigma', k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}.\sigma \geq \sigma'$. From the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, we know that $\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$.

Case $\varphi \equiv \boxminus_{[c,d]}\alpha$

(**Soundness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$

We have to show that $\forall\langle\sigma, left, right\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}, \eta.\textbf{dom}(\sigma) \supseteq \chi_O \wedge \tau_i - \tau_{right} \leq d \wedge \forall\sigma', j \in [left, right].(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. Take any arbitrary $\langle\sigma, left, right\rangle$ in $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$ such that σ is defined. From the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$, we know that $\tau_i - \tau_r \leq d$. Again from the construction, we know that $\sigma, left, right\rangle \in T$ where $T = S_{\text{new}} \cup S_{\text{update}} \cup S_{\text{carry-over}}$.

Sub-Sub-Case $\langle\sigma, left, right\rangle \in S_{\text{new}}$

From construction, $left = right = i$ and $\sigma \in \textbf{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha)$. From applicable mode checking judgements we see that **ips** soundness is applicable. From **ips** soundness we know that $\textbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. From mode judgement we also know that $\chi_C = \chi_F = \{\}$ and hence $\textbf{dom}(\sigma) \supseteq \chi_O$. From **ips** soundness we know that $\forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma')$. From this we can say that $\forall\sigma', j.(i \leq j \leq i \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$ completing our proof.

Sub-Sub-Case $\langle\sigma, left, right\rangle \in S_{\text{update}}$

From construction we know that $right = i$ and $\exists\sigma_1, \sigma_2.(\sigma = \sigma_1 \bowtie \sigma_2 \wedge \sigma_1 \in \textbf{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha) \wedge \langle\sigma_2, left, i-1\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H})$. From applicable mode checking judgements we see that **ips** soundness is applicable. From **ips** soundness we know that $\textbf{dom}(\sigma_1) \supseteq \chi_C \cup \chi_F \cup \chi_O = \chi_O$ as $\chi_C = \chi_F = \{\}$. We also know that $\forall\sigma_1'.(\sigma_1' \geq \sigma_1 \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma_1'$. Moreover, $\langle\sigma, left, i-1\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$. We know that $\hat{\pi}$ is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ. From which we know that $\hat{\pi}$ is strongly consistent at $i-1$ with respect to $\mathcal{L}$, τ, and φ. From

$\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$ soundness we have $\texttt{dom}(\sigma_2) \supseteq \chi_O \wedge \forall \sigma_2', j.(\mathit{left} \leq j \leq i-1 \wedge \sigma_2' \geq \sigma_2 \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma_2')$.

As σ is defined and $\sigma = \sigma_1 \bowtie \sigma_2$, $\texttt{dom}(\sigma) = \texttt{dom}(\sigma_1 \bowtie \sigma_2) \supseteq \chi_O$. Moreover, $\sigma \geq \sigma_1$ and $\sigma \geq \sigma_2$. Thus, combining the soundness we have $\mathcal{L}, \tau, i, \eta \models \alpha\sigma$ and $\forall j.(\mathit{left} \leq j \leq i-1 \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma)$. Again for any $\sigma' \geq \sigma$, we have $\sigma' \geq \sigma_1$ and $\sigma' \geq \sigma_2$ hence completing out proof.

Sub-Sub-Case $\langle \sigma, \mathit{left}, \mathit{right} \rangle \in S_{\text{carry-over}}$

From construction we know that $\langle \sigma, \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$. We know that $\hat{\pi}$ is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ. From which we know that $\hat{\pi}$ is strongly consistent at $i-1$ with respect to $\mathcal{L}$, τ, and φ. From $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$ soundness we have $\texttt{dom}(\sigma) \supseteq \chi_O \wedge \forall \sigma', j.(\mathit{left} \leq j \leq \mathit{right} \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. This completes our proof.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall \sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}, \eta.\texttt{dom}(\sigma) \supseteq \chi_O \wedge \forall \sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$. Take any arbitrary $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$. From semantics $\mathcal{L}, \tau, i, \eta \models (\boxminus_{[c,d]}\alpha)\sigma' \iff \forall j.(j \leq i \wedge c \leq \tau_i - \tau_j \leq d \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. Again from semantics we can write, $\mathcal{L}, \tau, i, \eta \models (\boxminus_{[c,d]}\alpha)\sigma' \iff \forall j.(\texttt{minPosition}(\tau, i, c, d) \neq -1 \wedge \texttt{maxPosition}(\tau, i, c, d) \neq -1 \wedge \texttt{minPosition}(\tau, i, c, d) \leq j \leq \texttt{maxPosition}(\tau, i, c, d) \leq i \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. From construction of $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$ implies that $\exists L, R.(\texttt{minPosition}(\tau, i, c, d) \neq -1 \wedge \texttt{maxPosition}(\tau, i, c, d) \neq -1 \wedge L \leq \texttt{minPosition}(\tau, i, c, d) \leq \texttt{maxPosition}(\tau, i, c, d) \leq R \wedge \langle \sigma, L, R \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H})$. From soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$ we have $\texttt{dom}(\sigma) \supseteq \chi_O$, $\tau_i - \tau_R \leq d$, and $\forall \sigma', j.(L \leq j \leq R \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. As $L \leq \texttt{minPosition}(\tau, i, c, d)$ and $\texttt{maxPosition}(\tau, i, c, d) \leq R$, we can write $\forall \sigma', j.(\texttt{minPosition}(\tau, i, c, d)$ $-1 \wedge \texttt{maxPosition}(\tau, i, c, d) \neq -1 \wedge \texttt{minPosition}(\tau, i, c, d) \leq j \leq \texttt{minPosition}(\tau, i, c, d) \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma')$. From this and semantics, we have our desired result of $\mathcal{L}, \tau, i, \eta \models (\boxminus_{[c,d]}\alpha)\sigma'$.

(**Completeness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$

We have to show that $\forall \sigma, L, R, \eta.\texttt{dom}(\sigma) \supseteq fv(\alpha) \wedge (L \leq R \leq i) \wedge (\tau_i - \tau_R \leq d) \wedge (\forall t.(L \leq t \leq R) \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma) \rightarrow \exists \sigma', \mathit{left}, \mathit{right}.(\sigma \geq \sigma') \wedge \mathit{left} \leq L \leq R \leq \mathit{right} \leq i \wedge \langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$. Take any arbitrary σ, L, and R such that $\texttt{dom}(\sigma) \supseteq fv(\alpha)$, $L \leq R \leq i$, $\tau_i - \tau_R \leq d$, and $\forall t.(L \leq t \leq R) \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma$.

Sub-Sub-Case $L < i, R = i$

We know: $\forall t.(L \leq t \leq R \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma)$. We can rewrite the above to : (a) $\forall t.(L \leq t \leq i-1 \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma)$. and (b) $\mathcal{L}, \tau, i, \eta \models \alpha\sigma$.

From (b) we see that $\texttt{ips}$ completeness is applicable. From $\texttt{ips}$ completeness we have, $\exists \sigma_2' \in \texttt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha).\sigma \geq \sigma_2'$. From construction $\sigma_2' \in \Sigma$.

From (a), we see that $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$ completeness is applicable. From $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$ completeness we have, $\exists L', \sigma_1'.(\sigma \geq \sigma_1' \wedge L' \leq L \leq (i-1) \wedge \langle \sigma_1', L', i-1 \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$. We know $\sigma \geq \sigma_1'$ and $\sigma \geq \sigma_2'$, so $\sigma_1' \bowtie \sigma_2'$ exists, and $\sigma \geq \sigma_1' \bowtie \sigma_2'$. We know $\langle \sigma_1', L', i-1 \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$, $\sigma_2' \in \Sigma$, and $\sigma_1' \bowtie \sigma_2' \neq \sigma_\perp$, from construction $\langle \sigma_1' \bowtie \sigma_2', L', i \rangle \in S_{\text{update}}$. We also know $L' \leq L \leq R \leq i$. From premise we know that $\tau_i - \tau_R \leq d$ hence $\langle \sigma_1' \bowtie \sigma_2', L', i \rangle$ not in S_{remove} hence in $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$.

Sub-Sub-Case $L = R = i$

We know $\mathcal{L}, \tau, i, \eta \models \alpha\sigma$. We see that **ips** completeness is applicable. From **ips** completeness we have, $\exists \sigma' \in \texttt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \alpha).(\sigma \geq \sigma')$. By construction $\sigma' \in \Sigma$. Case decision on: $\forall \langle \sigma_1, l, i-1 \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}.\sigma' \bowtie \sigma_1 \neq \sigma'$.

Sub-Sub-Sub-Case True

Then $\langle \sigma', i, i \rangle \in S_{\text{new}}$, so $\langle \sigma', i, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$, as $t_i - t_i \leq d$.

Sub-Sub-Sub-Case False

Then $\exists \langle \sigma_2, l, i-1 \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$ such that $\sigma' \bowtie \sigma_2 = \sigma'$. Then $\langle \sigma' \bowtie \sigma_2, l, i \rangle = \langle \sigma', l, i \rangle \in S_{\text{update}}$. As $t_i - t_i \leq d$, hence $\langle \sigma', l, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$.

Sub-Sub-Case $L < i, R < i$

Sub-Sub-Sub-Case $R = i - 1$

From premise we know that $\forall t.(L \leq t \leq R) \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma$. $\forall t.(L \leq t \leq i-1) \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma$. From the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$, we have $\exists \sigma', \mathit{left}, \mathit{right}.(\mathit{left} \leq L \leq R \leq \mathit{right} \leq (i-1) \wedge \sigma \geq \sigma' \wedge \langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H})$. We also know that $\tau_i - \tau_R \leq d$ and $R \leq \mathit{right}$. Thus, $\tau_i - \tau_{\mathit{right}} \leq d$.

Case decision on $\forall \sigma_1 \in \Sigma.\sigma_1 \bowtie \sigma \neq \sigma$.

Case: True: In case the above is condition is true, by construction, $\langle \sigma, \mathit{left}, \mathit{right} \rangle \in S_{\text{carry-over}}$ but $\langle \sigma, \mathit{left}, \mathit{right} \rangle \notin S_{\text{remove}}$. Hence, $\langle \sigma, \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$.

Case: False: In case the above is condition is false, it means there exists a $\sigma_1 \in \Sigma$ such that $\sigma \bowtie \sigma_1 = \sigma$. According to the construction, we see that $\langle \sigma, \mathit{left}, i \rangle \in S_{\text{update}}$ and $\tau_i - \tau_i \leq d$ ensures that $\langle \sigma, \mathit{left}, i \rangle \notin S_{\text{remove}}$, hence $\langle \sigma, \mathit{left}, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$, completing our proof.

Sub-Sub-Sub-Case $R < i - 1$

From premise we know that $\forall t.(L \leq t \leq R) \rightarrow \mathcal{L}, \tau, t, \eta \models \alpha\sigma$. Thus it implies that $\forall t.(L \leq t \leq i-1) \wedge \mathcal{L}, \tau, t, \eta \models \alpha\sigma$. From the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H}$, we have $\exists \sigma', \mathit{left}, \mathit{right}.(\mathit{left} \leq L \leq R \leq \mathit{right} < (i-1) \wedge \sigma \geq \sigma' \wedge \langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{H})$. We also know that $\tau_i - \tau_R \leq d$ and $R \leq \mathit{right}$. Thus, $\tau_i - \tau_{\mathit{right}} \leq d$. By construction, $\langle \sigma', \mathit{left}, \mathit{right} \rangle \in S_{\text{carry-over}}$ but $\langle \sigma', \mathit{left}, \mathit{right} \rangle \notin S_{\text{remove}}$ and hence $\langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$ which completes our proof.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall \sigma, \eta.\texttt{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, i, \eta \models \varphi\sigma \rightarrow (\exists \sigma' \in \pi.\mathcal{A}(\varphi)(i).\mathbb{R}.\sigma \geq \sigma')$. Take any arbitrary σ such that $\texttt{dom}(\sigma) \supseteq fv(\varphi)$ and $\mathcal{L}, \tau, i, \eta \models \boxminus_{[c,d]}\alpha\sigma$. Let $nP \leftarrow \texttt{minPosition}(\tau, i, c, d)$ and $xP \leftarrow \texttt{maxPosition}(\tau, i, c, d)$. From semantics we have: $\mathcal{L}, \tau, i, \eta \models \boxminus_{[c,d]}\alpha\sigma \iff \forall j.(j \leq i \wedge (c \leq \tau_i - \tau_j \leq d) \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma)$. From this we can also write: $\mathcal{L}, \tau, i, \eta \models \boxminus_{[c,d]}\alpha\sigma \iff \forall j.(nP \neq -1 \wedge xP \neq -1 \wedge (nP \leq j \leq xP) \rightarrow \mathcal{L}, \tau, j, \eta \models \alpha\sigma)$. From definition of **maxPosition**, we have $\tau_i - \tau_{xP} \leq d$. We see that completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$ is applicable. From the $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$ completeness we know that $\exists \sigma', \mathit{left}, \mathit{right}.(\sigma \geq \sigma' \wedge \mathit{left} \leq np \leq xP \leq \mathit{right} \leq i \wedge \langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H})$. As $\mathit{left} \leq np \leq xP \leq \mathit{right} \leq i$ and $\langle \sigma', \mathit{left}, \mathit{right} \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{H}$, from the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, we see that $\sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, hence completing our proof.

Case $\varphi \equiv \alpha \, \mathcal{S}_{[c,d]} \beta$

(**Soundness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$

The proof is exactly like the soundness proof of structure $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$ for $\diamondminus_{[c,d]}\alpha$.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$

We have to show that $\forall\langle\sigma,k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha, \eta.\mathbf{dom}(\sigma) \supseteq (\chi_1 \cup \chi_2) \wedge (\forall\sigma', l.(k \leq l \leq i) \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma')$. Take any arbitrary $\langle\sigma,k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$. From construction $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha \leftarrow S_{\text{new}} \cup S_{\text{update}}$. Thus, $\langle\sigma,k\rangle \in S_{\text{new}}$ or $\langle\sigma,k\rangle \in S_{\text{update}}$.

Sub-Sub-Case $\langle\sigma,k\rangle \in S_{\text{new}}$

By construction $k = i$ and $\sigma \in \bigcup_{\langle\sigma_\beta,j\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta \wedge j \neq i} \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_\beta, \alpha)$. From premise and soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, we know that $\mathbf{dom}(\sigma_\beta) \supseteq \chi_1$. Again from investigating the applicable mode checking judgements and soundness of $\mathbf{ips}$, $\mathbf{dom}(\sigma) \supseteq \chi_1 \cup \chi_2$. Again from soundness of $\mathbf{ips}$, we additionally know that, $\forall\sigma'.\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma'$. We also trivially satisfy that $\forall l.i \leq l \leq i$. Hence, we have our desired conclusion that $\forall\sigma', l.(k \leq l \leq i) \wedge \sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma'$.

Sub-Sub-Case $\langle\sigma,k\rangle \in S_{\text{update}}$

By construction we have $k < i$. By construction we also have $\exists\sigma_a, \sigma_1.\sigma = \sigma_a \bowtie \sigma_1 \wedge \sigma \neq \sigma_\perp \wedge \langle\sigma_a, k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha \wedge \sigma_1 \in \bigcup_{\langle\sigma_\beta,j\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta \wedge j \neq i} \mathbf{ips}(\mathcal{L}, i, \tau, \pi, \sigma_\beta, \alpha)$. From the premise, we have π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ. From definition of weak consistency, we can conclude that π is strongly consistent at $(i-1)$ with respect to $\mathcal{L}$, τ, and φ. From soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha$ we have, $\mathbf{dom}(\sigma_a) \supseteq (\chi_1 \cup \chi_2) \wedge \forall\sigma'', l''.(k \leq l'' \leq (i-1) \wedge \sigma'' \geq \sigma_a \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma'')$. Again from investigating the applicable mode checking judgements and soundness of $\mathbf{ips}$, $\mathbf{dom}(\sigma_1) \supseteq \chi_1 \cup \chi_2$ and $\forall\sigma'''.(\sigma''' \geq \sigma_1 \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma''')$. As $\sigma = \sigma_a \bowtie \sigma_1$ and $\sigma \neq \sigma_\perp$, $\mathbf{dom}(\sigma) = \mathbf{dom}(\sigma_a \bowtie \sigma_1) \supseteq (\chi_1 \cup \chi_2)$. Combining the above two soundness statements and using the fact that for any arbitrary $\sigma' \geq \sigma$ implies that $\sigma' \geq \sigma_a$ and $\sigma' \geq \sigma_1$, we have our desired result.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}, \eta.\mathbf{dom}(\sigma) \supseteq \chi_O \wedge \forall\sigma'.(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi\sigma')$. Take any arbitrary σ such that $\sigma \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$. From construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, $\sigma \in S_{R_1}$ or $\sigma \in S_{R_2}$.

Sub-Sub-Case $\sigma \in S_{R_1}$

From construction of S_{R_1}, $S_{R_1} \leftarrow \{\sigma_\beta \mid \langle\sigma_\beta, i\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta \wedge c \leq 0 \leq d\}$. From soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ we know that $\forall\langle\sigma_\beta, k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta.\mathbf{dom}(\sigma_\beta) \supseteq \chi_1 \wedge \tau_i - \tau_k \leq d \wedge \forall\sigma''.(\sigma'' \geq \sigma_\beta \rightarrow \mathcal{L}, \tau, k, \eta \models \beta\sigma'')$. We know $k = i$ and $\chi_O = \chi_1$. From the semantics of $\mathcal{S}$, we know that $\mathcal{L}, \tau, i, \eta \models \beta \wedge (c \leq 0 \leq d) \rightarrow \mathcal{L}, \tau, i, \eta \models (\alpha\,\mathcal{S}_{[c,d]}\beta)$. As $\sigma \in S_{R_1}$ the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ applies to σ. From construction we have $c \leq 0 \leq d$. Thus we have $\mathbf{dom}(\sigma) \supseteq \chi_1$. From the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ and semantics of $\mathcal{S}$, we have $\forall\sigma'(\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, i, \eta \models (\alpha\,\mathcal{S}_{[c,d]}\beta)\sigma'$.

Sub-Sub-Case $\sigma \in S_{R_2}$

From construction there exists σ_β, σ_α, k, j such that $\sigma = \sigma_\beta \bowtie \sigma_\alpha$, $\sigma \neq \sigma_\perp$, $\langle\sigma_\beta, k\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, $k \neq i$, $c \leq \tau_i - \tau_k \leq d$, $\langle\sigma_\alpha, j\rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, and $j \leq (k+1)$. From the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, we have $\mathbf{dom}(\sigma_\beta) \supseteq \chi_1 \wedge \tau_j - \tau_k \leq$

$d \wedge \forall \sigma''.(\sigma'' \geq \sigma_\beta \rightarrow \mathcal{L}, \tau, k, \eta \models \beta\sigma'')$. Again from the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, we have $\mathbf{dom}(\sigma_\alpha) \supseteq (\chi_1 \cup \chi_2) \wedge (\forall \sigma'', l.(k \leq l \leq i) \wedge \sigma'' \geq \sigma_\alpha \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma')$. We know $\chi_O = \chi_1$ and $\sigma = \sigma_\beta \bowtie \sigma_\alpha$ from which we have $\mathbf{dom}(\sigma) \supseteq (\chi_O \cup \chi_2)$. Thus, we have our desired $\mathbf{dom}(\sigma) \supseteq \chi_O$. From the semantics of $\mathcal{S}$ we have, $\mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[lo,hi]} \beta) \iff \exists m \leq i. \mathcal{L}, \tau, m, \eta \models \beta \wedge lo \leq \tau_i - \tau_m \leq hi \wedge \forall t.m+1 \leq l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha$. From the semantics, combining the soundness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ and $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, by instantiating $m = k$ and $t = l$, and using the fact that for any arbitrary σ' such that $\sigma' \geq \sigma$ implies that $\sigma' \geq \sigma_\beta$ and $\sigma' \geq \sigma_\alpha$, we have the following $\mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[c,d]} \beta)\sigma'$.

(**Completeness**)

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$

The proof is exactly like the completeness proof of the structure $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{P}$ of $\diamondminus_{[c,d]}\alpha$.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$

Take any arbitrary σ, σ_β, and k such that $\langle \sigma_\beta, k \rangle \in \mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, $\sigma \geq \sigma_\beta$, $\mathbf{dom}(\sigma) \supseteq (\chi_1 \cup \chi_2)$, and $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma)$. We have to show that $\exists \sigma', m.(\sigma \geq \sigma' \wedge m \leq k+1 \wedge \langle \sigma', m \rangle \in \mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha)$.

Sub-Sub-Case $k < (i-1)$

We know that $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma)$. From this we can write $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma) \rightarrow \forall l.(k < l \leq (i-1) \wedge \mathcal{L}, \tau, i, \eta \models \alpha\sigma)$. We know that π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ. From this we know that π is strongly consistent at $i-1$ with respect to $\mathcal{L}$, τ, and φ. From the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha$ we know that $\exists \sigma_1, m.(\sigma \geq \sigma_1 \wedge m \leq k+1 \wedge \langle \sigma_1, m \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha)$. We also can write $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma) \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma$. From investigating applicable mode judgements we see that the **ips** completeness is applicable. From i.h. completeness of **ips** we know that $\exists \sigma_2.(\sigma \geq \sigma_2 \wedge \sigma_2 \in \mathbf{ips}(\mathcal{L}, i, \tau, \hat{\pi}, \sigma_\beta, \alpha)$. We see from the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$ that $\sigma_2 \in \Sigma_\alpha$. So far we have $\sigma \geq \sigma_1$ and $\sigma \geq \sigma_2$. From this we know that $\sigma_1 \bowtie \sigma_2$ is defined and $\sigma \geq \sigma_1 \bowtie \sigma_2$. By construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, $\sigma_1 \bowtie \sigma_2 \in S_{\text{update}}$ hence $\sigma_1 \bowtie \sigma_2 \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$.

Sub-Sub-Case $k = (i-1)$

We know that $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma)$. We also can write $\forall l.(k < l \leq i \wedge \mathcal{L}, \tau, l, \eta \models \alpha\sigma) \rightarrow \mathcal{L}, \tau, i, \eta \models \alpha\sigma$. From investigating applicable mode judgements we see that the **ips** completeness is applicable. From i.h. completeness of **ips** we know that $\exists \sigma_2.(\sigma \geq \sigma_2 \wedge \sigma_2 \in \mathbf{ips}(\mathcal{L}, i, \tau, \hat{\pi}, \sigma_\beta, \alpha)$. We see from the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$ that $\sigma_2 \in \Sigma_\alpha$. Now we will show that σ_2 is either in S_{new} or in S_{update} and hence in $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$.

Sub-Sub-Sub-Case $\forall \langle \sigma_1, t \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha.(\sigma_1 \bowtie \sigma_2 \neq \sigma_2)$

In which case from the construction, the side condition for $\sigma_2 \in S_{\text{new}}$ is true and consequently $\sigma_2 \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$.

Sub-Sub-Sub-Case $\exists \langle \sigma_1, t \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha.(\sigma_1 \bowtie \sigma_2 = \sigma_2)$

From this we can write $\exists \langle \sigma_1, t \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha.(\sigma_1 \bowtie \sigma_2 = \sigma_2) \rightarrow \exists \langle \sigma_1, t \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha.(\sigma_1 \bowtie \sigma_2 = \sigma_\perp)$. Moreover, we have $\sigma_1 \bowtie \sigma_2 = \sigma_2$. From the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, we see that $\sigma_2 \in S_{\text{update}}$ and hence $\sigma_2 \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$, completing our proof.

Sub-Case $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$

We have to show that $\forall \sigma, \eta. \mathbf{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, \tau, i, \eta \models \varphi\sigma \rightarrow (\exists \sigma' \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}.\ \sigma \geq \sigma')$. Take any arbitrary σ such that $\mathbf{dom}(\sigma) \supseteq fv(\varphi)$ and $\mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[c,d]} \beta)\sigma$. From the semantics of $\mathcal{S}$ we have $\mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[c,d]} \beta)\sigma \iff \exists k.((k \leq i) \wedge \mathcal{L}, \tau, k, \eta \models \beta\sigma \wedge (c \leq \tau_i - \tau_k \leq d) \wedge \forall l.(k < l \leq i \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma))$. We have the following two cases.

Sub-Sub-Case $k = i$

From the semantics of $\mathcal{S}$, we have $\mathcal{L}, \tau, i, \eta \models \beta\sigma \wedge c \leq \tau_i - \tau_i \leq d \rightarrow \mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[c,d]} \beta)\sigma$. From premise we have $\mathbf{dom}(\sigma) \supseteq fv(\varphi)$ from which we know $\mathbf{dom}(\sigma) \supseteq fv(\varphi) \supseteq fv(\beta)$. We also have $\tau_i - \tau_i \leq d$. We see the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ is applicable. We thus have $\mathcal{L}, \tau, i, \eta \models \beta\sigma \rightarrow \exists \sigma_1.(\langle \sigma_1, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta \wedge (\sigma \geq \sigma_1))$. From construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, $\langle \sigma_1, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, and $c \leq 0 \leq d$, we have $\sigma_1 \in S_{R_1}$ and hence $\sigma_1 \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, completing our proof.

Sub-Sub-Case $k < i$

From the semantics of $\mathcal{S}$ we have, $\mathcal{L}, \tau, k, \eta \models \beta\sigma \wedge (c \leq \tau_i - \tau_k \leq d) \wedge \forall l.((k < l \leq i) \rightarrow \mathcal{L}, \tau, l, \eta \models \alpha\sigma) \rightarrow \mathcal{L}, \tau, i, \eta \models (\alpha\, \mathcal{S}_{[c,d]} \beta)\sigma$. From premise we have $\mathbf{dom}(\sigma) \supseteq fv(\varphi)$ from which we know $\mathbf{dom}(\sigma) \supseteq fv(\varphi) \supseteq fv(\beta)$. We also have $\tau_i - \tau_k \leq d$. We see the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$ is applicable. We thus have $\mathcal{L}, \tau, i, \eta \models \beta\sigma \wedge \tau_i - \tau_k \leq d \rightarrow \exists \sigma_\beta.(\langle \sigma_\beta, i \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta \wedge (\sigma \geq \sigma_\beta))$.

Investigating applicable mode checking judgements and using the Lemma 1 we have, $\chi_1 \subseteq fv(\beta)$ and $\chi_2 \subseteq fv(\alpha)$. We also know from the definition of the function fv, $fv(\alpha\, \mathcal{S}_{[c,d]} \beta) = fv(\alpha) \cup fv(\beta)$. Thus, we have $\chi_1 \cup \chi_2 \subseteq fv(\alpha) \cup fv(\beta) = fv(\alpha\, \mathcal{S}_{[c,d]} \beta) \supseteq \mathbf{dom}(\sigma)$. We see that the completeness of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$ is applicable. From which we have $\exists \sigma_\alpha, m. \sigma \geq \sigma_\alpha \wedge m \leq (k+1) \wedge \langle \sigma_\alpha, m \rangle \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha$.

As we have $\sigma \geq \sigma_\beta$ and $\sigma \geq \sigma_\alpha$, $\sigma_\beta \bowtie \sigma_\alpha$ is defined and $\sigma \geq \sigma_\beta \bowtie \sigma_\alpha$. From the construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, we know that $\sigma_\beta \bowtie \sigma_\alpha \in S_{R_2}$ and hence $\sigma_\beta \bowtie \sigma_\alpha \in \hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, completing our proof.

Proof of part (2): Mutual induction on the structure of φ. We show select cases and other cases are similar.

Case $\varphi \equiv \top$.

(**Soundness**)

From definition of **ips**, $\mathbf{ips}(\mathcal{L}, j, \pi, \sigma_{\text{in}}, \top) = \Sigma_{out} = \{\sigma_{\text{in}}\}$. From premise 1 and from mode checking judgement [TRUE], $\chi_C, \chi_F \vdash \top : \emptyset$. From premise 3, $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F$. From above, for all $\sigma \in \Sigma_{out}$, $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$ as $\chi_O = \emptyset$. We have to show that $\forall \sigma'. \sigma' \geq \sigma \wedge \mathcal{L}, \tau, j, \eta_0 \models \top\sigma'$. From the semantics, any $\sigma' \geq \sigma$ trivially satisfy $\mathcal{L}, \tau, j, \eta_0 \models \top\sigma'$.

(**Completeness**)

Let $\sigma_o = \sigma_{in}$. Then by premise $\sigma_o \leq \sigma$, and by definition of **ips** $\sigma_o \in \Sigma_{out}$.

Case $\varphi \equiv \bot$.

(**Soundness**)

From definition of **ips**, $\mathbf{ips}(\mathcal{L}, j, \pi, \sigma_{\text{in}}, \bot) = \Sigma_{out} = \emptyset$. Thus the statement is vacuously true.

(**Completeness**)

For any σ, $\bot\sigma = \bot$. Since there are no $\mathcal{L}$, τ, η_0, and j such that $\mathcal{L}, \tau, j, \eta_0 \models \bot$, the statement is vacuously true.

Case $\varphi \equiv \mathsf{p}(t_1, \ldots, t_n)$.
(**Soundness**)

From definition of $\texttt{ips}$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \mathsf{p}(t_1, \ldots, t_n)) = \Sigma_{out} = \texttt{sat}(\mathcal{L}, j$ $, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}})$. From premise 1 and 3, pre-condition of the $\texttt{sat}$ function is satisfied (Claim 1). From Claim 1, for all $\sigma \in \texttt{sat}(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots,$ $t_n), \sigma_{\text{in}})$, $\texttt{dom}(\sigma) = \chi_C \cup \chi_F \cup \chi_O$. Thus, we can write, for all $\sigma \in \Sigma_{out}$, $\texttt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. It remains to show $\forall \sigma'.\sigma' \geq \sigma \rightarrow \mathcal{L}, \tau, j, \eta_0 \models \mathsf{p}(t_1, \ldots, t_n)\sigma'$. From Claim 1, for all $\sigma \in \texttt{sat}(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}})$, $\mathcal{L}, \tau, j, \eta_0 \models \mathsf{p}(t_1, \ldots, t_n)\sigma$. Note that, the function $\texttt{sat}$ returns grounding substitutions[1] for $\mathsf{p}(t_1, \ldots, t_n)$. Thus, by Corollary 1 for all $\sigma' \geq \sigma$ where $\sigma \in \texttt{sat}(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{\text{in}})$, $\mathcal{L}, \tau, j, \eta_0 \models \mathsf{p}(t_1, \ldots, t_n)\sigma'$ holds.

(**Completeness**)

Let $V = fv(\mathsf{p}(t_1, \ldots, t_n))$. By the semantics of predicates, it must be that $\texttt{dom}(\sigma) \supseteq V$. Then by premise and Lemma 11, $\mathcal{L}, \tau, j, \eta_0 \models \mathsf{p}(t_1, \ldots, t_n)[\sigma \downarrow V]$. Let $\sigma_o = \sigma \downarrow (V \cup \texttt{dom}(\sigma_{in}))$. Since $\sigma_{in} \leq \sigma$ by premise, $\sigma \downarrow V \leq \sigma_o \leq \sigma$. By Corollary 1, it follows that $\mathcal{L}, \tau, j, \eta_0 \models \mathsf{p}(t_1, \ldots, t_n)\sigma_o$.

Case $\varphi \equiv \varphi_1 \vee \varphi_2$.
(**Soundness**)

Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)$ and $\Sigma_2 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_2)$. From definition of $\texttt{ips}$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \vee \varphi_2) = \Sigma_{out} = \Sigma_1 \cup \Sigma_2$. Then $\sigma \in \Sigma_1$ or $\sigma \in \Sigma_2$. W.l.o.g., $\sigma \in \Sigma_1$. By inspection of disjunction mode judgements (and Lemmas 5, 7, and 8), $\chi_C, \chi_F \vdash \varphi_1 : \chi_1$. By I.H., $\texttt{dom}(\sigma) \supseteq (\chi_C \cup \chi_F \cup \chi_1)$ and $\forall \sigma'.\sigma' \geq \sigma \implies \mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma'$. Since $\chi_O = \chi_1 \cap \chi_2$, $\texttt{dom}(\sigma) \supseteq (\chi_C \cup \chi_F \cup \chi_1) \supseteq (\chi_C \cup \chi_F \cup \chi_O)$. Further, by semantics of $\vee$, $\forall \sigma''.L, \tau, j, \eta_0 \models \varphi_1\sigma'' \implies L, \tau, j, \eta_0 \models (\varphi_1 \vee \varphi_2)\sigma''$. Thus, $\forall \sigma'.\sigma' \geq \sigma \implies \mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \vee \varphi_2)\sigma'$, which concludes soundness.

(**Completeness**)

If $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \vee \varphi_2)\sigma$, then $\mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma$ or $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma$. W.l.o.g., $\mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma$. Since $fv(\varphi_1) \subseteq fv(\varphi_1 \vee \varphi_2)$, by I.H. there exists $\sigma_o \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ such that $\sigma_o \leq \sigma$. By definition of $\texttt{ips}$, $\sigma_o \in \Sigma_{out}$.

Case $\varphi \equiv \varphi_1 \wedge \varphi_2$.
(**Soundness**)

From the definition of $\texttt{ips}$,
$\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \wedge \varphi_2) = \bigcup_{\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)} \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_1)$. Take an arbitrary $\sigma \in \Sigma_{out}$. Then there exists $\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ such that $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. By inspection of the applicable mode checking judgements (and Lemmas 7, 5), the inductive hypothesis is applicable and yields $\texttt{dom}(\sigma_c) \supseteq \chi_C \cup \chi_F \cup \chi_1$. Now, with the additional help of Lemma 8 the inductive hypothesis yields $\texttt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_1 \cup \chi_2$. Since $\chi_O = \chi_1 \cup \chi_2$, $\texttt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$.

[1] Substitutions for all free variables

It remains to show that $\forall\sigma'.(\sigma' \geq \sigma \rightarrow (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \wedge \varphi_2)\sigma'))$. Take any arbitrary σ' such that $\sigma' \geq \sigma$. By inductive hypothesis on φ_2 we have $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma'$. Further $\sigma' \geq \sigma_c$, since $\sigma \geq \sigma_c$ by Lemma 12. Then we can apply the inductive hypothesis and get $\mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma'$. From the semantics of $\wedge$, we have $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \wedge \varphi_2)\sigma'$.

(**Completeness**)

If $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \wedge \varphi_2)\sigma$, then $\mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma$ and $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma$. Since $fv(\varphi_i) \subseteq fv(\varphi_1 \wedge \varphi_2)$, the I.H. is applicable and guarantees $\exists\sigma_i \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_i).\sigma_i \leq \sigma$. By Lemma 12, also $\sigma_i \geq \sigma_{in}$, so $\mathbf{dom}(\sigma_i) \supseteq \chi_C \cup \chi_F$. Let $\sigma_2' = \sigma \downarrow \mathbf{dom}(\sigma_1) \cup \mathbf{dom}(\sigma_2) \cup fv(\varphi_2)$, which is a prefix of σ and $\mathbf{dom}(\sigma_2') = \mathbf{dom}(\sigma_1) \cup \mathbf{dom}(\sigma_2) \cup fv(\varphi_2)$, since $\mathbf{dom}(\sigma_i) \subseteq \mathbf{dom}(\sigma)$ and $\mathbf{dom}(\sigma) \supseteq fv(\varphi) \supseteq fv(\varphi_2)$. So $\sigma_i \leq \sigma_2' \leq \sigma$. By inductive hypothesis for soundness, since $\sigma_2' \geq \sigma_2$, $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma_2'$. Thus we can apply the inductive hypothesis with $\sigma_{in} = \sigma_1$ and get $\exists\sigma_o \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_1, \varphi_2).\sigma_o \leq \sigma_2'$. By definition of $\mathbf{ips}$ $\sigma_o \in \Sigma_{out}$, and by transitivity $\sigma_o \leq \sigma$.

Case $\varphi \equiv \exists\vec{x}.\varphi$.

(**Soundness**)

W.l.o.g., we have $\mathbf{dom}(\sigma_{in}) \cap \{\vec{x}\} = \emptyset$ as we can rename $\vec{x}$ to some fresh $\vec{y}$. Let $\sigma \in \Sigma_{out}$. By definition, $\mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \exists\vec{x}.\varphi) = \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}},$
$\varphi) \setminus \{\vec{x}\}$. Thus there exists a $\vec{t}$ and a $\sigma_1 \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi)$ such that $\sigma[\vec{x} \mapsto \vec{t}] = \sigma_1$. In other words, $\sigma = \sigma_1 \setminus \{\vec{x}\}$. By inspection of the mode checking judgements, we can apply the inductive hypothesis, which yields $\mathbf{dom}(\sigma_1) \supseteq \chi_C \cup \chi_F \cup \chi_1$ and $\forall\sigma'' \geq \sigma_1.\mathcal{L}, \tau, j, \eta_0 \models \varphi_1\sigma''$.

From $\sigma = \sigma_1 \setminus \{\vec{x}\}$, it follows that $\mathbf{dom}(\sigma) = \mathbf{dom}(\sigma_1) \setminus \{\vec{x}\}$. From $\mathbf{dom}(\sigma_1) \supseteq \chi_C \cup \chi_F \cup \chi_1$, we have $\mathbf{dom}(\sigma_1) \setminus \{\vec{x}\} \supseteq (\chi_C \cup \chi_F \cup \chi_1) \setminus \{\vec{x}\} = (\chi_C \setminus \{\vec{x}\}) \cup (\chi_F \setminus \{\vec{x}\}) \cup (\chi_1 \setminus \{\vec{x}\})$. From $\mathbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$ and $\mathbf{dom}(\sigma_{in}) \cap \{\vec{x}\} = \emptyset$, we have $\mathbf{dom}(\sigma_1) \setminus \{\vec{x}\} \supseteq \chi_C \cup \chi_F \cup (\chi_1 \setminus \{\vec{x}\}) = \chi_C \cup \chi_F \cup \chi_O$. Finally we have, $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$.

Take any arbitrary σ' such that $\sigma' \geq \sigma$. We can write $\sigma' + [\vec{x} \mapsto \vec{t}] \geq \sigma + [\vec{x} \mapsto \vec{t}] = \sigma_1$. From i.h., $\mathcal{L}, \tau, j, \eta_0 \models \varphi(\sigma' + [\vec{x} \mapsto \vec{t}])$. From semantics of $\exists$, we can write $\mathcal{L}, \tau, j, \eta_0 \models (\exists\vec{x}.\varphi)\sigma'$.

(**Completeness**)

$\mathcal{L}, \tau, j, \eta_0 \models (\exists\vec{x}.\varphi)\sigma$ if and only if there exists a $\vec{t}$ such that $\mathcal{L}, \tau, j, \eta_0 \models \varphi(\sigma + [\vec{x} \mapsto \vec{t}])$. From premise we have $\sigma \geq \sigma_{in}$ and from which it follows that $\sigma + [\vec{x} \mapsto \vec{t}] \geq \sigma_{in}$. From the definition of fv we have $fv(\varphi) \subseteq fv(\exists\vec{x}.\varphi) \cup \{\vec{x}\}$. From premise we also have $\mathbf{dom}(\sigma) \supseteq fv(\exists\vec{x}.\varphi)$. $\mathbf{dom}(\sigma) \cup \{\vec{x}\} \supseteq fv(\exists\vec{x}.\varphi) \cup \{\vec{x}\} \supseteq fv(\varphi)$. Thus, we can write $\mathbf{dom}(\sigma + [\vec{x} \mapsto \vec{t}]) \supseteq fv(\varphi)$ as $\mathbf{dom}(\sigma + [\vec{x} \mapsto \vec{t}]) = \mathbf{dom}(\sigma) \cup \{\vec{x}\}$. We now see that the i.h. is applicable, from which we have, $\exists\sigma_{ol} \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi)$ such that $\sigma + [\vec{x} \mapsto \vec{t}] \geq \sigma_{ol}$. From which we can write $\sigma \geq \sigma_{ol} \setminus \{\vec{x}\}$. From the definition of $\mathbf{ips}$, $\sigma_{ol} \setminus \{\vec{x}\} \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists\vec{x}.\varphi)$ which completes our proof.

Case $\varphi \equiv \forall\vec{x}.(\varphi_1 \rightarrow \varphi_2)$.

(**Soundness**)

Consider, $\mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \forall\vec{x}.(\varphi_1 \rightarrow \varphi_2)) \neq \emptyset$. In which case, $\mathbf{ips}(\mathcal{L}, j, \tau,$
$\pi, \sigma_{\text{in}}, \forall\vec{x}.(\varphi_1 \rightarrow \varphi_2)) = \{\sigma_{\text{in}}\}$. Consider any σ_1 such that $\sigma_1 \in \mathbf{ips}(\mathcal{L}, j, \tau,$
$\pi, \sigma_{\text{in}}, \varphi_1)$ and σ_2 such that $\sigma_2 \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_1, \varphi_2)$. If no such σ_1 exists then φ is trivially satisfied, by falsifying the antecedent, in which case $\mathbf{ips}$ returns σ_{in} and thus from premise $\mathbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F\chi_O$ where $\chi_O = \{\}$. If no such σ_2 exists then φ is falsified the statement

vacuously holds. Now consider both σ_1 and σ_2 exists. From premise 3, we know $\texttt{dom}(\sigma_{\text{in}}) \supseteq \chi_C \cup \chi_F$. We also know $\sigma_{\text{in}} \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \forall\vec{x}.(\varphi_1 \to \varphi_2))$. From mode checking judgements we know, $\chi_O = \emptyset$. Thus, for $\sigma \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \forall\vec{x}.(\varphi_1 \to \varphi_2))$, $\texttt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$.

It remains to show that $\forall\sigma'.(\sigma' \geq \sigma \to (\mathcal{L}, \tau, j, \eta_0 \models (\forall\vec{x}.(\varphi_1 \to \varphi_2))\sigma'))$. We know from the definition $\sigma = \sigma_{\text{in}}$ as $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \forall\vec{x}.(\varphi_1 \to \varphi_2)) \neq \emptyset$. Take any arbitrary σ' such that $\sigma' \geq \sigma = \sigma_{in}$.

Inspection of the applicable mode checking judgements reveals that in all cases $fv(\phi_2) \subseteq \chi_C \cup \chi_F \cup \chi_1$ (with transitivity and additivity of $\subseteq$). From premise (1) and Lemma 1 or Lemma 2, $\chi_1 \subseteq fv(\phi_1)$. Thus $fv(\phi_2) \subseteq \chi_C \cup \chi_F \cup fv(\phi_1)$. Then always $fv(\phi_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\}$, so that $fv(\phi_2) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\}$. Finally, by premise 3 we know $\texttt{dom}(\sigma_{\text{in}}) \supseteq (\chi_C \cup \chi_F)$, which means: (**C-i**) $fv(\phi_2) \subseteq \texttt{dom}(\sigma_{in}) \cup \{\vec{x}\}$.

$\mathcal{L}, \tau, j, \eta_0 \models (\forall\vec{x}.(\varphi_1 \to \varphi_2))\sigma'$ is equivalent to $\forall\vec{t}.(\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1)(\sigma'[\vec{x} \mapsto \vec{t}]) \to \mathcal{L}, \tau, j, \eta_0 \models (\varphi_2)(\sigma'[\vec{x} \mapsto \vec{t}]))$. Take any arbitrary $\vec{t}$ such that $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1)(\sigma'[\vec{x} \mapsto \vec{t}])$.

[**Z**] *It is thus sufficient to show that*: $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_2)(\sigma'[\vec{x} \mapsto \vec{t}])$.

By I.H. (Completeness), $\exists\sigma_1 \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1).\sigma_1 \leq \sigma'[\vec{x} \mapsto \vec{t}]$. By inspection of the mode checking judgements and I.H. (Soundness), $\texttt{dom}(\sigma_1) \supseteq \texttt{dom}(\sigma_{in}) \cup \{\vec{x}\}$. From construction, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_1, \varphi_2) \neq \emptyset$. Take an arbitrary σ_2 from this set. By I.H. (Soundness), (**C-ii**) $\sigma_2 \geq \sigma_1 \wedge \mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma_2$.

Now, we will show that: $[\mathcal{TS}]$ $\exists\sigma_2^m \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_1, \varphi_2).(\sigma_2 \geq \sigma_2^m \wedge \texttt{dom}(\sigma_2^m) \subseteq \texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\} \wedge \mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma_2^m)$. From (**C-ii**), we have (A-II) $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma_2$. From Lemma 11 and A-II, we have (A-III) $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2(\sigma_2 \downarrow fv(\varphi_2))$.

From Lemma 11, we have (A-IV) $\forall\sigma, \sigma', \varphi.((\texttt{dom}(\sigma) = fv(\varphi) \wedge \texttt{dom}(\sigma) \cap \texttt{dom}(\sigma') = \emptyset \wedge \mathcal{L}, \tau, j, \eta_0 \models \varphi\sigma) \to (\mathcal{L}, \tau, j, \eta_0 \models \varphi[\sigma + \sigma']))$. From (**C-i**), we have (A-V) $fv(\varphi_2) \subseteq \texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\}$. It follows that: $\exists Y.(fv(\varphi_2) \cup Y) = \texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\}$. From (A-III), (A-IV), and (A-V), we have (A-VI) $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2(\sigma_2 \downarrow (\texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\}))$. (X) By I.H. (Completeness), $\exists\sigma_2^m \leq (\sigma_2 \downarrow (\texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\})).\sigma_2^m \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_1, \varphi_2(\vec{x}))$. As $\sigma_2^m \geq \sigma_2^m$, and by I.H. (Soundness), $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma_2^m$. Thus, we have shown the third conjunct of $\mathcal{TS}$ to be true. From (X), we know (Y) $\sigma_2^m \leq \sigma_2 \downarrow (\texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\})$. It implies that $\sigma_2^m \leq \sigma_2$. Thus, we have shown the first conjunct of $\mathcal{TS}$ to be true. From (A-V) and from (Y), $\texttt{dom}(\sigma_2^m) \subseteq \texttt{dom}(\sigma_2 \downarrow (\texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\})) = \texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\}$. Thus, we have shown the second conjunct of $\mathcal{TS}$ to be true. This implies that we have shown $\mathcal{TS}$ to be true.

Now, if we can show that $\sigma'[\vec{x} \mapsto \vec{t}] \geq \sigma_2^m$ then from the third conjunct of $\mathcal{TS}$ and I.H. (Soundness), we can show $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_2)(\sigma'[\vec{x} \mapsto \vec{t}])$ to hold. $\sigma' \geq \sigma_2^m[\vec{x} \mapsto \vec{t}]$ is equivalent to the following:

[**U**] $\forall v \in \texttt{dom}(\sigma_2^m).\sigma_2^m(v) = \sigma'[\vec{x} \mapsto \vec{t}](v)$.

From second conjunct of $\mathcal{TS}$, $\texttt{dom}(\sigma_2^m) \subseteq \texttt{dom}(\sigma_{\text{in}}) \cup \{\vec{x}\}$. From this and **U**, we can say that for all $v \in \texttt{dom}(\sigma_2^m)$, either (**E-1**) $v \in ((\texttt{dom}(\sigma_{\text{in}}) \setminus \{\vec{x}\}) \cap \texttt{dom}(\sigma_2^m))$ or (**E-2**) $v \in (\{\vec{x}\} \cap \texttt{dom}(\sigma_2^m))$ holds.

Sub-Case (**E-1**) $v \in ((\texttt{dom}(\sigma_{\text{in}}) \setminus \{\vec{x}\}) \cap \texttt{dom}(\sigma_2^m))$:

We know $\sigma' \geq \sigma_{\text{in}}$. It implies that $\sigma'(v) = \sigma_{\text{in}}(v)$. We also have $\texttt{dom}(\sigma') \supseteq \texttt{dom}(\sigma_{\text{in}})$.

Consider $v \notin \{\vec{x}\}$, so [**R-1**] $\sigma'[\vec{x} \mapsto \vec{t}](v) = \sigma_{\text{in}}(v)$. We know $\sigma_2 \geq \sigma_1 \geq \sigma_{\text{in}} \setminus \{\vec{x}\}$. We can write $\sigma_2 \geq \sigma_{\text{in}} \setminus \{\vec{x}\}$. This is equivalent to $\forall v_1 \in \mathtt{dom}(\sigma_{\text{in}}) \setminus \{\vec{x}\}.\sigma_2(v_1) = \sigma_{\text{in}} \setminus \{\vec{x}\}(v_1)$. We also know from the first conjunct of $\mathcal{TS}$ that $\sigma_2 \geq \sigma_2^m$. It is equivalent to $\forall v_2 \in \mathtt{dom}(\sigma_2^m).\sigma_2(v_2) = \sigma_2^m(v_2)$. As $v \in ((\mathtt{dom}(\sigma_{\text{in}}) \setminus \{\vec{x}\}) \cap \mathtt{dom}(\sigma_2^m))$, $v \in \mathtt{dom}(\sigma_{\text{in}}) \setminus \{\vec{x}\}$ and $v \in \mathtt{dom}(\sigma_2^m)$. It implies that $\sigma_2^m(v) = \sigma_2(v) = \sigma_{\text{in}} \setminus \{\vec{x}\}(v)$. As $v \notin \{\vec{x}\}$, it implies that $\sigma_2^m(v) = \sigma_{\text{in}}(v)$. From above and **R-1**, we have $\sigma_2^m(v) = \sigma'[\vec{x} \mapsto \vec{t}](v)$.

Sub-Case (**E-2**) $v \in (\{\vec{x}\} \cap \mathtt{dom}(\sigma_2^m))$:

We have to show that $\sigma_2^m(v) = \sigma'[\vec{x} \mapsto \vec{t}](v)$. We know $\sigma_2 \geq \sigma_2^m$ which implies that $\forall v_1 \in \mathtt{dom}(\sigma_2^m).\sigma_2(v_1) = \sigma_2^m(v_1)$. As $\sigma_2^m \in \mathtt{ips}(\mathcal{L}, j, \pi, \sigma_1, \varphi_2(\vec{x}))$, we have $\sigma_2^m \geq \sigma_1$. It implies that $\forall v_2 \in \mathtt{dom}(\sigma_1).\sigma_2^m(v_2) = \sigma_1(v_2)$. We also know $\sigma'[\vec{x} \mapsto \vec{t}] \geq \sigma_1$ which implies that $\forall v_3 \in \mathtt{dom}(\sigma_1).\sigma_1(v_3) = \sigma'[\vec{x} \mapsto \vec{t}](v_3)$. By inspecting the mode checking judgements we know $\{\vec{x}\} \subseteq \chi_1$. Thus, we know $\mathtt{dom}(\sigma_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\}$. As $v \in \{\vec{x}\}$, it implies that $v \in \mathtt{dom}(\sigma_1)$. Thus, we have $\forall v \in \{\vec{x}\}.\sigma_2^m(v) = \sigma_1(v) = \sigma'[\vec{x} \mapsto \vec{t}](v)$ completing our proof.

(**Completeness**)

We have to show that $\forall \sigma.(\sigma \geq \sigma_{in} \wedge \mathtt{dom}(\sigma) \supseteq fv(\varphi) \wedge \mathcal{L}, i, \tau, \eta \models \varphi\sigma \rightarrow \exists \sigma_o(\sigma_o \in \mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)) \rightarrow \sigma \geq \sigma_o)$. Take any arbitrary σ such that $\sigma \geq \sigma_{in}$, $\mathtt{dom}(\sigma) \supseteq fv(\varphi)$, and $\mathcal{L}, i, \tau, \eta \models \varphi\sigma$. W.l.o.g we assume $\mathtt{dom}(\sigma) \cap \vec{x} = \{\}$ and $\mathtt{dom}(\sigma_{in}) \cap \vec{x} = \{\}$. From semantics we know that, $\mathcal{L}, \tau, i, \eta \models (\forall \vec{x}.(\varphi_1 \rightarrow \varphi_2))\sigma \iff \forall \vec{t}.\mathcal{L}, \tau, i, \eta \models \varphi_1\sigma[\vec{x} \mapsto \vec{t}] \rightarrow \mathcal{L}, \tau, i, \eta \models \varphi_2\sigma[\vec{x} \mapsto \vec{t}]$. Take any arbitrary σ' from $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \varphi_1)$. By Lemma 12, $\sigma' \geq \sigma_{in}$. By Lemma 13, $\mathtt{dom}(\sigma') \subseteq \mathtt{dom}(\sigma_{in}) \cup fv(\varphi_1)$. By analyzing mode checking judgements: $fv(\varphi_1) \subseteq \chi_C \cup \chi_F \cup \{\vec{x}\} \subseteq \mathtt{dom}(\sigma_{in}) \cup \{\vec{x}\}$ (by premise $\chi_C \cup \chi_F \supseteq \mathtt{dom}(\sigma_{in})$). By analyzing applicable mode checking judgements and ips soundness (IH) we have : $\vec{x} \subseteq \mathtt{dom}(\sigma')$. By premise, $\sigma \geq \sigma_{in}$ and $fv(\varphi) \subseteq \mathtt{dom}(\sigma)$ and $\{\vec{x}\} \cap \mathtt{dom}(\sigma) = \{\}$. Again, $fv(\varphi) = (fv(\varphi_1) \cup fv(\varphi_2)) \setminus \{\vec{x}\}$. Let $\sigma'' = \sigma + [\vec{x} \mapsto \sigma'(\vec{x})]$. We will now show that $\sigma'' \geq \sigma'$. From the definition of $\geq$, $\sigma'' \geq \sigma'$ if for any $v \in \mathtt{dom}(\sigma').\sigma''(v) = \sigma'(v)$. We have seen that $\mathtt{dom}(\sigma') \subseteq \mathtt{dom}(\sigma_{in}) \cup \{\vec{x}\}$. Thus for any $v \in \mathtt{dom}(\sigma')$, $v \in \mathtt{dom}(\sigma_{in})$ or $v \in \{\vec{x}\}$.

Sub-Case $v \in \mathtt{dom}(\sigma_{in})$

From Lemma 12, $\sigma' \geq \sigma_{in}$. From premise $\sigma \geq \sigma_{in}$. $\sigma' \geq \sigma_{in} \rightarrow \sigma'(v) = \sigma_{in}(v)$. $\sigma \geq \sigma_{in} \rightarrow \sigma(v) = \sigma_{in}(v) \rightarrow \sigma(v) = \sigma''(v) = \sigma'(v)$.

Sub-Case $v \in \{\vec{x}\}$

$v \in \vec{x} \rightarrow \sigma'(v) = \sigma''(v)$ by construction as no other variables in domain of σ'.

Thus we have $\sigma'' \geq \sigma'$. By analyzing the applicable mode checking judgements we see that **ips** soundness is applicable. From **ips** soundness we have $\mathcal{L}, i, \tau, \eta \models \varphi_1\sigma'' = \mathcal{L}, i, \tau, \eta \models \varphi_1\sigma[\vec{x} \mapsto \sigma'(\vec{x})]$. Let us also assume $\vec{t} \leftarrow \sigma'(\vec{x})$. By premise and the semantics of universal quantifier: $\mathcal{L}, i, \tau, \eta \models \varphi_2\sigma[\vec{x} \mapsto \vec{t}]$. From mode checking judgements we know that $\chi_C, \chi_F \cup \chi_1 \vdash \varphi_2 : \chi_2$. From premise we have $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi) \geq \Delta(\varphi_2)$. From **ips** soundness of φ_1, we know $\mathtt{dom}(\sigma') \supseteq \chi_C \cup \chi_F \cup \chi_1$. From premise we know that π is consistent at i with respect to $\mathcal{L}$, τ, and φ (φ_2). We have shown $\sigma \geq \sigma_{in}$. We also have $fv(\varphi_2) \subseteq \mathtt{dom}(\sigma'')$ as $fv(\varphi_2) \subseteq \chi_C \cup \chi_F \cup \chi_1 \subseteq \mathtt{dom}(\sigma') \subseteq \mathtt{dom}(\sigma'')$. We also have $\mathcal{L}, i, \tau, \eta \models \varphi_2\sigma[\vec{x} \mapsto \vec{t}]$. Hence we see that **ips** completeness is applicable. By **ips** completeness (IH) we have $\exists \sigma_2' \in \mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma', \varphi_2).\sigma_2' \leq \sigma[\vec{x} \mapsto \vec{t}]$. Thus, $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma', \varphi_2) \neq \{\}$. Thus, there does not exist a $\sigma \in \mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \varphi_1)$ such that $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma, \varphi_2) = \{\}$.

Thus, $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{in}, \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)) = \sigma_{in}$. Now by premise $\sigma_{in} \leq \sigma$.

Case $\varphi \equiv \varphi_1 \mathcal{S}_{[c,d]} \varphi_2$.

(**Soundness**)

Sub-Case $\mathbf{B} \in \mathit{label}(\varphi)$

From the definition of $\mathtt{ips}$, we get $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2) = \sigma_{\text{in}} \bowtie \pi.\mathcal{A}(\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)(j).\mathbb{R}$. We can say that for all $\sigma \in \mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \mathcal{S} \varphi_2)$ where $\sigma \neq \emptyset$, there exists a $\sigma_1 \in \pi.\mathcal{A}(\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)(j).\mathbb{R}$ such that $\sigma = \sigma_{\text{in}} \bowtie \sigma_1$. We have to first show that for all $\sigma \in \mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2)$, $\mathtt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. From premise 3, Lemma 14 (1), and Definition 7, $\mathtt{dom}(\sigma_{\text{in}}) \supseteq \chi_C \cup \chi_F$ and $\mathtt{dom}(\sigma_1) \supseteq \chi_O$. As $\sigma = \sigma_{\text{in}} \bowtie \sigma_1$ and $\sigma \neq \emptyset$, we can see that $\mathtt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. It remains to show that $\forall \sigma'.(\sigma' \geq \sigma \rightarrow (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'))$. Take any arbitrary, σ' such that $\sigma' \geq \sigma$. As $\sigma = \sigma_{\text{in}} \bowtie \sigma_1$ and $\sigma \neq \emptyset$, we can write $\sigma' \geq \sigma_1$. From Lemma 14 (1) and Definition 7, we know that $\forall \sigma_1'.\sigma_1' \geq \sigma_1 \rightarrow \mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma_1'$.

It follows that $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'$, completing the proof.

Sub-Case $\mathbf{B} \notin \mathit{label}(\varphi)$

Take an arbitrary $\sigma \in \Sigma_{out}$. By definition, $\Sigma_{out} = \mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2) = S_{R_1} \cup S_{R_2}$. Thus, $\sigma \in S_{R_1}$ or $\sigma \in S_{R_2}$.

Sub-Sub-Case $\sigma \in S_{R_1}$

From definition of S_{R_1}, we know that $\sigma \in \mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_2)$ and $c \leq 0 \leq d$. We first show that $\mathtt{dom}(\sigma) \supseteq (\chi_C \cup \chi_F \cup \chi_O)$. From premise 3, we know that $\mathtt{dom}(\sigma_{\text{in}}) \supseteq (\chi_C \cup \chi_F)$. From the mode checking judgements for $\mathcal{S}$ and I.H., $\mathtt{dom}(\sigma) \supseteq (\chi_C \cup \chi_F \cup \chi_1)$. Since $\chi_O = \chi_1$ by the applicable judgements, $\mathtt{dom}(\sigma) \supseteq (\chi_C \cup \chi_F \cup \chi_O)$. It remains to show that $\forall \sigma'.(\sigma' \geq \sigma \rightarrow (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'))$. Take an arbitrary σ' such that $\sigma' \geq \sigma$. From the semantics of $\mathcal{S}$ we know that, $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'$ if and only if there exists $k \in \mathbb{N}$ and $k \leq j$ such that $(c \leq \tau_j - \tau_k \leq d)$ and $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2\sigma'$ and for all $l \in \mathbb{N}$ such that $k < l \leq j$, it implies that $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1\sigma'$ holds. So if $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma'$ holds and $c \leq 0 \leq d$, then $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'$ holds. Now from construction, since $\sigma \in \mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$, $c \leq 0 \leq d$, and $\sigma' \geq \sigma$, by inductive hypothesis it follows that $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_2)\sigma'$. From this, it follows that $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'$.

Sub-Sub-Case $\sigma \in S_{R_2}$

Then there exist $\langle \sigma_\beta, k \rangle \in S_\beta$ and σ_l^α, such that $c \leq \tau_j - \tau_k \leq d$, $\bowtie \sigma_l^\alpha = \sigma$ and $k < j$ and for all l with $k < l \leq j$ we have $\sigma_l^\alpha \in \mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \varphi_1)$. For brevity, from here on we assume l is sufficiently restricted to the domain of σ_l^α. By construction, $\sigma_\beta \in \mathtt{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, $\mathtt{dom}(\sigma_\beta) \supseteq \chi_C \cup \chi_F \cup \chi_1$, and since $\chi_O = \chi_1$, $\mathtt{dom}(\sigma_\beta) \supseteq \chi_C \cup \chi_F \cup \chi_O$. Now by Lemma 12, $\forall l.\sigma_l^\alpha \geq \sigma_\beta$. Thus, $\forall l.\mathtt{dom}(\sigma_l^\alpha) \supseteq \chi_C \cup \chi_F \cup \chi_O$, and so $\mathtt{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$.

It remains to show that $\forall \sigma'.(\sigma' \geq \sigma \rightarrow (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma'))$. Take any arbitrary, σ' such that $\sigma' \geq \sigma$. Then $\sigma' \geq \sigma_\beta$, so by inductive hypothesis $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2\sigma'$ and also $c \leq \tau_j - \tau_k \leq d$. Also $\forall l.\sigma' \geq \sigma_l^\alpha$, so that again by inductive hypothesis $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1\sigma'$. The semantics of $\mathcal{S}$ is $\mathcal{L}, \tau, i, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma' \Leftrightarrow \exists m \in \mathbb{N}.(m \leq i \wedge \mathcal{L}, \tau, m, \eta_0 \models \varphi_2\sigma' \wedge (c \leq \tau_j - \tau_m \leq d) \wedge \forall l \in \mathbb{N}.((m < l \leq i) \rightarrow \mathcal{L}, \tau, i, \eta_0 \models \alpha\sigma'))$. Instantiation of m with k and i with j lets us conclude.

(**Completeness**)

$\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)\sigma$ if and only if $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1\sigma) \mathcal{S}_{[c,d]} (\varphi_2\sigma)$ if and only if there exists $k \leq j$ such that $c \leq \tau_j - \tau_k \leq d$, $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2\sigma$ and for all l, where $k < l \leq j$, $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1\sigma$. Let k be maximal.

Sub-Case $\mathbf{B} \in \mathit{label}(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$

Since $\mathbf{B} \in \mathit{label}(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$, there exist $\chi_C^{\mathbf{B}}$, $\chi_O^{\mathbf{B}}$ with $\mathbf{dom}(\chi_O^{\mathbf{B}}) \subseteq fv(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$, $\chi_1^{\mathbf{B}} = \chi_O^{\mathbf{B}}$ and $\chi_2^{\mathbf{B}}$, such that $\emptyset \vdash_{\mathbf{B}} \varphi_1 : \chi_1^{\mathbf{B}}$, $\chi_1^{\mathbf{B}} \vdash_{\mathbf{B}} \varphi_2 : \chi_2^{\mathbf{B}}$ and thus $\chi_C^{\mathbf{B}} \vdash_{\mathbf{B}} \varphi_1 \mathcal{S} \varphi_2 : \chi_O^{\mathbf{B}}$. Let $\sigma' = \sigma \downarrow \chi_O^{\mathbf{B}}$. Note that $\mathbf{dom}(\sigma') = \chi_O^{\mathbf{B}}$, since $\mathbf{dom}(\sigma) \supseteq fv(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$. Since π is strongly consistent at i with respect to $\mathcal{L}$, τ, and φ and $j \leq i$, π is well-formed at j with respect to $\varphi_1 \mathcal{S}_{[c,d]}\varphi_2$, $\mathcal{L}$, and τ ($\Psi(\mathcal{L}, \tau, \pi, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2, j)$). So, by Definition 6 (Statement 6 of $\varphi_1 \mathcal{S}_{\mathbb{I}}\varphi_2$) $\sigma' \in \pi.\mathcal{A}(\varphi_1 \mathcal{S}_{\mathbb{I}}\varphi_2)(j)$.
$\mathbb{R}$. Let $\sigma_o = \sigma_{in} \bowtie \sigma'$. Note that $\sigma_o \neq \emptyset$, because σ' is a prefix of σ, which itself is an extension of σ_{in}. Thus $\sigma_o \in \Sigma_{out}$. By the same arguments also $\sigma_o \leq \sigma$.

Sub-Case $\mathbf{B} \notin \mathit{label}(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$

Sub-Sub-Case $k = j$.

Since $fv(\varphi_2) \subseteq fv(\varphi_1 \mathcal{S}_{[c,d]}\varphi_2)$, by inductive hypothesis $\exists \sigma_o \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2).\sigma_o \leq \sigma$. By construction, $\langle \sigma_o, j\rangle$ in S_β. As $j = k$, $\tau_j - \tau_k = 0$, so from premise $c \leq 0 \leq d$. Thus $\sigma_o \in S_{R_1}$ and so $\sigma_o \in \Sigma_{out}$.

Sub-Sub-Case $k < j$.

Then analogous to the previous case $\exists \sigma_2 \in \mathbf{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2).\sigma_2 \leq \sigma$. From premise we have $c \leq \tau_j - \tau_k \leq d$. Also, for all l such that $j \geq l > k$ $\sigma_2 \notin \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$, or k would not be maximal for σ. Thus, $\langle \sigma_2, k\rangle \in S_{\varphi_2}$. By inspection of the mode checking judgements (and lemmas) and soundness, $\mathbf{dom}(\sigma_2) \supseteq \chi_C \cup \chi_O \cup \chi_1$. Thus, by I.H. for all l with $k < l \leq j$, $\exists \sigma_l^\alpha \in \mathbf{ips}(\mathcal{L}, l, \tau, \pi, \sigma_2, \varphi_1).\sigma_l^\alpha \leq \sigma$.

Since all σ_l^α are $\leq \sigma$, the join $\sigma_o = \bowtie \sigma_l^\alpha$ exists and $\sigma_o \leq \sigma$. Furthermore, by construction $\sigma_o \in S_{R_2}$ and so $\sigma_o \in \Sigma_{out}$.

Case $\varphi \equiv \varphi_1 \mathcal{U}_{[c,d]}\varphi_2$.

(**Soundness**)

Take an arbitrary $\sigma \in \Sigma_{out}$. By definition, $\Sigma_{out} = \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_1 \mathcal{U}_{[c,d]} \varphi_2) = S_{R_1} \cup S_{R_2}$. Thus, $\sigma \in S_{R_1}$ or $\sigma \in S_{R_2}$.

Sub-Case $\sigma \in S_{R_1}$

From definition of S_{R_1}, we know $S_{R_1} = \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_2)$ and $c \leq 0 \leq d$, so $\sigma \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_2)$. We first show that $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. From premise 3, we know that $\mathbf{dom}(\sigma_{\mathrm{in}}) \supseteq (\chi_C \cup \chi_F)$. From the mode checking judgements for UNTIL and I.H., $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_1$. Since $\chi_O = \chi_1$ by the applicable judgements, $\mathbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$.

It still remains to show that $\forall \sigma'.(\sigma' \geq \sigma \rightarrow (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]}\varphi_2)\sigma'))$. Take any arbitrary, σ' such that $\sigma' \geq \sigma$. From the semantics of $\mathcal{U}$ we know that, $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]}\varphi_2)\sigma'$ if and only if there exists $k \in \mathbb{N}$ where $k \geq j$ and $c \leq (\tau_k - \tau_j) \leq d$, such that $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2\sigma'$ and for all $l \in \mathbb{N}$ such that $j \leq l < k$, it implies that $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1\sigma'$ holds. So if $\mathcal{L}, \tau, j, \eta_0 \models \varphi_2\sigma'$ and $c \leq 0 \leq d$ holds, then $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]}\varphi_2)\sigma'$ holds. Now since $\sigma \in \mathbf{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\mathrm{in}}, \varphi_2)$, by I.H. it follows that $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_2)\sigma'$, and so $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]}\varphi_2)\sigma'$.

Sub-Case $\sigma_1 \in S_{R_2}$

Then there exist $\langle \sigma_\beta, k\rangle \in S_{\varphi_2}$ and σ_l^α, such that $\bowtie \sigma_l^\alpha = \sigma$ and $k > j$ and $\forall l.(j \leq l < k \to \sigma_l^\alpha \in \texttt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_\beta, \varphi_1)$. For brevity, again assume l restricted. By construction, $\sigma_\beta \in \texttt{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2)$ and $c \leq (\tau_k - \tau_j) \leq d$. From premise 3, we know that $\textbf{dom}(\sigma_{\text{in}}) \supseteq \chi_C \cup \chi_F$. From the mode checking judgements for UNTIL and I.H., $\textbf{dom}(\sigma_\beta) \supseteq (\chi_C \cup \chi_F \cup \chi_1)$. Since $\chi_O = \chi_1$ by the applicable judgements, $\textbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. Now by Lemma 12, $\forall l.\sigma_l^\alpha \geq \sigma_\beta$. Thus, $\forall l.\textbf{dom}(\sigma_l^\alpha) \supseteq \chi_C \cup \chi_F \cup \chi_O$, and so $\textbf{dom}(\sigma) \supseteq \chi_C \cup \chi_F \cup \chi_O$. It remains to show that $\forall \sigma'.(\sigma' \geq \sigma \to (\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]} \varphi_2)\sigma'))$. Take any arbitrary, σ' such that $\sigma' \geq \sigma$. Then $\sigma' \geq \sigma_\beta$, so by inductive hypothesis $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2 \sigma'$. Also $\forall l.\sigma' \geq \sigma_l^\alpha$, so that again by inductive hypothesis $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1 \sigma'$. The semantics of $\mathcal{U}$ is $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]} \varphi_2)\sigma'$ if and only if there exists $m \in \mathbb{N}$ where $m \geq j$ and $c \leq (\tau_m - \tau_j) \leq d$, such that $\mathcal{L}, \tau, m, \eta_0 \models \varphi_2 \sigma'$ and for all $l \in \mathbb{N}$ such that $j \leq l < m$, it implies that $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1 \sigma'$ holds. Instantiation of m with k lets us conclude.

(**Completeness**)

$\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \mathcal{U}_{[c,d]} \varphi_2)\sigma$ if and only if $\mathcal{L}, \tau, j, \eta_0 \models (\varphi_1 \sigma) \mathcal{U}_{[c,d]} (\varphi_2 \sigma)$ if and only if there exists $k \geq j$, such that $c \leq \tau_k - \tau_j \leq d$, $\mathcal{L}, \tau, k, \eta_0 \models \varphi_2 \sigma$, and for all l where $j \leq l < k$ $\mathcal{L}, \tau, l, \eta_0 \models \varphi_1 \sigma$. W.l.o.g. k is minimal.

Sub-Case $k = j$

Since $fv(\varphi_2) \subseteq fv(\varphi_1 \mathcal{U}_{[c,d]} \varphi_2)$, by inductive hypothesis $\exists \sigma_o \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2).\sigma_o \leq \sigma$. Since $c \leq 0 \leq d$ and construction of S_{φ_2}, $\langle \sigma_o, j\rangle$ in S_{φ_2}, thus $\sigma_o \in S_{R_1}$ and so $\sigma_o \in \Sigma_{out}$.

Sub-Case $k > j$

Then analogous to the previous case $\exists \sigma_2 \in \texttt{ips}(\mathcal{L}, k, \tau, \pi, \sigma_{in}, \varphi_2).\sigma_2 \leq \sigma$. Also, for all $l < k$ $\sigma_2 \notin \texttt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$, or k would not be minimal for σ. Thus, $\langle \sigma_2, k\rangle \in S_{\varphi_2}$. By $\sigma \geq \sigma_{in}$, $\textbf{dom}(\sigma) \supseteq \textbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$. By inspection of the mode checking judgements (and lemmas) and soundness, $\textbf{dom}(\sigma_2) \supseteq \chi_C \cup \chi_O \cup \chi_1$. Thus, by I.H. for all l with $j \leq l < k$, $\exists \sigma_l^\alpha \in \texttt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_2, \varphi_1).\sigma_l^\alpha \leq \sigma$. Since all σ_l^α are $\leq \sigma$, the join $\sigma_o = \bowtie \sigma_l^\alpha$ exists and $\sigma_o \leq \sigma$. Furthermore, by construction $\sigma_o \in S_{R_2}$ and so $\sigma_o \in \Sigma_{out}$.

□

The following lemma states that the number of substitutions returned by **ips** is finite. It also states that the size of our state π (which stores the summary structures) is also finite. The following lemma is used to show the termination of **ips**, **uSS**, and consequently **checkCompliance**.

Lemma 15 (Finite substitutions). *1. For all formulas φ of form either $\varphi_1 \mathcal{S}_{\mathbb{I}} \varphi_2$, $\diamondminus_{\mathbb{I}} \varphi$, $\boxminus_{\mathbb{I}} \varphi$, or $\ominus_{\mathbb{I}} \varphi$, such that $\boldsymbol{B} \in label(\varphi)$, for all $i \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all state $\pi = (\mathcal{A}, i)$ such that π is weakly consistent at i with respect to $\mathcal{L}$, τ, and φ, if $\left(\sum_{\hat{\varphi} \in \textsf{b-s-tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right) + \Upsilon(\pi, i-1, \varphi)$ is finite then $\sum_{\hat{\varphi} \in \textsf{b-tsub}(\varphi)} \Upsilon(\hat{\pi}, i, \hat{\varphi})$ is finite where $\hat{\pi} = (\mathcal{A}, i)$ and $\hat{\pi} = \textbf{uSS}(\mathcal{L}, i, \tau, \pi, \varphi)$.*

2. For all formula φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all state $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for all substitution σ_{in}, for some given χ_C and χ_F, such that: (1) $\chi_C, \chi_F \vdash \varphi : \chi_O$, (2) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, (3) $\textbf{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$, (4) π

is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$, (5) $\left(\sum_{\hat{\varphi}\in \mathsf{b\text{-}tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right)$ is finite, if $\boldsymbol{\texttt{ips}}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi) = \Sigma_{out}$ *then* $|\Sigma_{out}|$ *is finite.*

Proof. Note that, when **uSS** is called for a **B-formula** φ, it calls **ips** on strict subformulas of φ. However, **ips** does not call **uSS** directly and hence we do not have any cyclic dependency. Mutual induction on the structure of φ. We show select cases. Other cases are similar.

Case $\varphi \equiv \top$.

According to the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \top) = \Sigma_{out} = \{\sigma_{in}\}$. Thus, $|\Sigma_{out}| = 1$ and is thus finite.

Case $\varphi \equiv \bot$.

According to the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \bot) = \Sigma_{out} = \emptyset$. Thus, $|\Sigma_{out}| = 0$ and is thus finite.

Case $\varphi \equiv \mathsf{p}(t_1, \ldots, t_n)$.

According to the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \mathsf{p}(t_1, \ldots, t_n)) = \Sigma_{out} = \texttt{sat}(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{in})$. From premise (1) we can say the pre-condition of the **sat** function (Claim 1) is satisfied. From Claim 1, we can say that $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \varphi_1 \vee \varphi_2$.

Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ and $\Sigma_2 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$. According to the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \varphi_1 \vee \varphi_2) = \Sigma_{out} = \Sigma_1 \cup \Sigma_2$. From premise 1 and inspecting the mode checking judgements, we can say the inductive hypothesis is applicable. By inductive hypothesis, $|\Sigma_1|$ and $|\Sigma_2|$ are both finite. Thus, $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \varphi_1 \wedge \varphi_2$.

From the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \wedge \varphi_2) = \Sigma_{out}$ where $\Sigma_{out} = \bigcup_{\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)} \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. From premise 1 and inspecting the mode checking judgements, we see that the inductive hypothesis is applicable to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)$. Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)$. By inductive hypothesis, $|\Sigma_1|$ is finite. For all $\sigma_c \in \Sigma_1$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$ is called. We also see that from premise 1 and inspecting the mode checking judgements, the inductive hypothesis is applicable to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2) = \Sigma_2$ for some $\sigma_c \in \Sigma_1$. By inductive hypothesis, each such $|\Sigma_2|$ is finite from which it follows that $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \exists \vec{x}.\varphi$.

Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists \vec{x}.\varphi)$ According to the definition of **ips**, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists \vec{x}.\varphi) = \Sigma_{out} = \Sigma_1 \setminus \{\vec{x}\}$. From premise 1 and inspecting mode checking judgements, we see the induction hypothesis is applicable. By inductive hypothesis, $|\Sigma_1|$ is finite from which it follows that $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$.

Let $\Sigma_{out} \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2))$. Σ_{out} can be either $\emptyset$ or $\{\sigma_{in}\}$ according to the definition of **ips**. In both cases, $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \varphi_1 \mathcal{S}_{[c,d]} \varphi_2$ and $\mathbf{B} \in \mathit{label}(\varphi)$

Sub-Case To show (1):

Given $\left(\sum_{\hat{\varphi} \in \mathsf{b\text{-}s\text{-}tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right) + \Upsilon(\pi, i-1, \varphi)$ is finite to show that $\sum_{\hat{\varphi} \in \mathsf{b\text{-}tsub}(\varphi)} \Upsilon(\hat{\pi}, i, \hat{\varphi})$ is finite, it is sufficient to show that $\sum_{\hat{\varphi} \in \mathsf{b\text{-}tsub}(\varphi)} \Upsilon(\hat{\pi}, i, \hat{\varphi}) - \left(\sum_{\hat{\varphi} \in \mathsf{b\text{-}s\text{-}tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right) - \Upsilon(\pi, i-1, \varphi)$ is finite. From the definition of Υ (Definition 9), we know that: $\sum_{\hat{\varphi} \in \mathsf{b\text{-}tsub}(\varphi)} \Upsilon(\hat{\pi}, i, \hat{\varphi}) - \left(\sum_{\hat{\varphi} \in \mathsf{b\text{-}s\text{-}tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right) - \Upsilon(\pi, i-1, \varphi) = (|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha| + |\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta)| + |\pi.\mathcal{A}(\varphi)(i).\mathbb{R}|)$.
Thus, it is sufficient to show that $(|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha| + |\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta| + |\pi.\mathcal{A}(\varphi)(i).\mathbb{R}|)$ is finite. We will show that the following are all finite: $|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha|$, $|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta|$, and $|\pi.\mathcal{A}(\varphi)(i).\mathbb{R}|$.

By construction, $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta| \leq |\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\beta| + |\Sigma_\beta|$. From definition of Σ_β, we know that $\Sigma_\beta \leftarrow \mathtt{ips}(\mathcal{L}, i, \tau, \hat{\pi}, \bullet, \varphi_2)$. From premise, Lemma 4, and consulting the applicable mode checking judgements (and Lemma 5, 8), we see that the inductive hypothesis of (2) is applicable. By inductive hypothesis, $|\Sigma_\beta|$ is finite and let us assume it is m_1. From the premise, we know that $|\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\beta|$ is finite and let us assume it is m_2. Thus, $|\hat{\pi}.\mathcal{A}'(\varphi)(i).S_\beta| \leq m_1 + m_2$, which is also finite.

Again by construction, $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha| \leq (\times|\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha)| + |\Sigma_\alpha|)$. From the induction assumption, we know that $|\hat{\pi}.\mathcal{A}'(\varphi)(i-1).\mathbb{S}_\alpha)|$ is finite and thus to show $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha|$ is finite it is sufficient to show that $|\Sigma_\alpha|$ is finite.

We will now show that $|\Sigma_\alpha|$ is finite. To construct Σ_α, in the worst case, for all $\langle \sigma, k \rangle$ pairs in $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta$, $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma, \varphi_1)$ is called. From the premise, Lemma 4, and consulting the applicable mode checking judgements, we see that the inductive hypothesis of (2) is applicable. By inductive hypothesis of (2), each call to $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma, \varphi_1)$ returns a finite set of substitutions. Let us assume the maximum cardinality of, all the sets of substitutions returned by the calls to $\mathtt{ips}$, is m_3. Thus by construction, $|\Sigma_\alpha| \leq (m_1 + m_2) \times m_3$, which is finite. It follows that $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha|$ is finite.

From construction of $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}$, we know that $\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R} = S_{R_1} \cup S_{R_2}$. Thus, $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}| \leq |S_{R_1}| + |S_{R_2}|$. We will show that $|S_{R_1}|$ and $|S_{R_2}|$ are both finite, concluding our proof.

From construction, $|S_{R_1}| \leq |\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta|$. As $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta)|$ is finite (shown above), $|S_{R_1}|$ is finite.

From construction, $|S_{R_2}| \leq |\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta| \times |\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha)|$. As shown above, $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\beta|$ and $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{S}_\alpha|$ are both finite. This concludes our proof that $|S_{R_2}|$ is finite and in turn $|\hat{\pi}.\mathcal{A}'(\varphi)(i).\mathbb{R}|$ is finite.

Sub-Case To show (2):

Let $\Sigma_1 \leftarrow \pi.\mathcal{A}(\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{R}$. From the definition of $\mathtt{ips}$, $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2) = \Sigma_{out} = \sigma_{\text{in}} \bowtie \pi.\mathcal{A}(\varphi_1 \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{R}$. As we are given by the premise that $\left(\sum_{\hat{\varphi} \in \mathsf{b\text{-}tsub}(\varphi)} \Upsilon(\pi, i, \hat{\varphi})\right)$ is finite, we know that $|\Sigma_1|$ is finite. It follows that $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \varphi_1 \mathcal{S}_{[c,d]} \varphi_2$ and $\mathbf{B} \notin label(\varphi)$

According to the definition of $\mathtt{ips}$, $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \varphi_1 \mathcal{S}_{[c,d]} \varphi_2) = \Sigma_{out} = S_{R_1} \cup S_{R_2}$. It is sufficient to show that $|S_{R_1}|$ and $|S_{R_2}|$ are both finite.

We will first show that by inductive hypothesis, $|S_\beta|$ is finite. In the worst case, $c \leq \tau_j - \tau_0 \leq d$, that is all prior trace positions satisfy the interval constraint $[c, d]$.

Let us now consider, for all l where $0 \leq l \leq j$, $m = max|\Sigma_1|.\Sigma_1 \leftarrow \mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$. The maximum size of $|S_\beta|$ can be $(j + 1) \times m$ by construction. By the inductive hypothesis, m is finite. As $j \in \mathbb{N}$ is finite, it follows that $|S_\beta|$ is finite.

By construction, as $S_{R_1} \subseteq S_\beta$, $|S_{R_1}| \leq |S_\beta|$ and it follows that $|S_{R_1}|$ is finite.

By construction of S_{R_2}, for each $\langle \sigma_\beta, k \rangle \in S_\beta$ where $k \neq j$, $\mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta,$ $\varphi_1)$ is called for all q such that $k < q \leq j$. By inductive hypothesis, $|\mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta, \varphi_1)|$ is finite and let us say for all q such that $k < q \leq j$, $m_1 = max|\Sigma_2|.\Sigma_2 \leftarrow \mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta, \varphi_1)$. By construction ($\bowtie$ of all substitutions for all positions q), $|S_{R_2}|$ is finite.

Thus, it follows that $|\Sigma_{out}|$ is finite.

Case $\varphi \equiv \varphi_1 \mathcal{U}_{[c,d]} \varphi_2$.

According to the definition of $\mathtt{ips}$, $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \varphi_1 \mathcal{U}_{[c,d]} \varphi_2) = \Sigma_{out} = S_{R_1} \cup S_{R_2}$. It is sufficient to show that $|S_{R_1}|$ and $|S_{R_2}|$ are both finite.

We will first show that by inductive hypothesis, $|S_\beta|$ is finite. Let us consider, for all l where $b \leq l < e$, $l \geq j$, e is the minimal position such that $\tau_e - \tau_j > d$, b is the minimal position such that $c \leq \tau_b - \tau_j \leq d$, and $m = max|\Sigma_1|.\Sigma_1 \leftarrow \mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$. From premise 2, we can say that such a finite e exists. The maximum size of $|S_\beta|$ is less than $(e - b) \times m$ by construction. By the inductive hypothesis, m is finite. As $e \in \mathbb{N}$ is finite, it follows that $|S_\beta|$ is finite.

By construction, as $S_{R_1} \subseteq S_\beta$, $|S_{R_1}| \leq |S_\beta|$ and it follows that $|S_{R_1}|$ is finite.

By construction of S_{R_2}, for each $\langle \sigma_\beta, k \rangle \in S_\beta$ where $k > j$, $\mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta,$ $\varphi_1)$ is called for all q such that $j \leq q < k$. By inductive hypothesis, $|\mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta, \varphi_1)|$ is finite and let us say for all q such that $j \leq q < k$, $m_1 = max|\Sigma_2|.\Sigma_2 \leftarrow \mathtt{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta, \varphi_1)$. By construction ($\bowtie$ of all substitutions of all positions q), $|S_{R_2}|$ is finite.

Thus, it follows that $|\Sigma_{out}|$ is finite.

□

Lemma 16 (Termination). *For all formula φ, the following holds:*

1. *For all logs $\mathcal{L}$, for all time stamp sequences τ, for all $j \in \mathbb{N}$, for all substitution σ_{in}, for all state $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for some given χ_C, χ_F such that: (1) $\chi_C, \chi_F \vdash \varphi : \chi_O$, (2) $\mathtt{dom}(\sigma_{in}) \supseteq \chi_C \cup \chi_F$, (3) π is strongly consistent at i with respect to φ, τ, and $\mathcal{L}$, (4) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, then $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi)$ terminates.*

2. *For all logs $\mathcal{L}$, for all time stamp sequences τ, for all $i \in \mathbb{N}$, for all state $\pi = (\mathcal{A}, i)$ such that φ is either of form $\varphi_1 \mathcal{S}_{[c,d]} \varphi_2$, $\diamondminus_{[c,d]} \varphi$, $\boxminus_{[c,d]} \varphi$, or $\ominus_{[c,d]} \varphi$ such that $\emptyset \vdash_{\mathbf{B}} \varphi : \chi_O$, and π is weakly consistent at i with respect to φ and $\mathcal{L}$, then $\mathtt{uSS}(\mathcal{L}, i, \tau, \pi, \varphi)$ terminates.*

Proof. Mutual induction on the structure of φ. We show select cases and rest of the cases are similar.

Case $\varphi \equiv \top$.

$\texttt{ips}(\mathcal{L}, j, \pi, \sigma_{in}, \top)$ terminates trivially.

Case $\varphi \equiv \bot$.

$\texttt{ips}(\mathcal{L}, j, \pi, \sigma_{in}, \bot)$ terminates trivially.

Case $\varphi \equiv \mathsf{p}(t_1, \ldots, t_n)$.

According to the definition of $\texttt{ips}$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \mathsf{p}(t_1, \ldots, t_n)) = \Sigma_{out} =$ $\texttt{sat}(\mathcal{L}, j, \tau, \mathsf{p}(t_1, \ldots, t_n), \sigma_{in})$. From premise (1) we can say the pre-condition of the $\texttt{sat}$ function (Claim 1) is satisfied. From Claim 1, we can say that $\texttt{sat}$ terminates and from it follows that $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \mathsf{p}(t_1, \ldots, t_n))$ terminates.

Case $\varphi \equiv \varphi_1 \vee \varphi_2$.

According to the definition of, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1 \vee \varphi_2) = \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1) \cup$ $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$. From premise 1 and inspecting the mode checking judgements, we can say the inductive hypothesis is applicable. By inductive hypothesis, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ and $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$ both terminate. Moreover, by Lemma 15, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ and $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_2)$, each independently returns finite number of substitutions. Hence, taking the union of the substitutions terminate. It follows that $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1 \vee \varphi_2)$ terminates.

Case $\varphi \equiv \varphi_1 \wedge \varphi_2$.

From the definition of $\texttt{ips}$, $\texttt{ips}(\mathcal{L}, j, \pi, \sigma_{\text{in}}, \varphi_1 \wedge \varphi_2) = \Sigma_{out}$ where $\Sigma_{out} =$ $\bigcup_{\sigma_c \in \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)} \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. From premise 1 and inspecting the mode checking judgements, we see that the inductive hypothesis is applicable to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)$. By inductive hypothesis, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1)$ terminates. Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \pi, \sigma_{\text{in}}, \varphi_1)$. From Lemma 15, we have $|\Sigma_1|$ is finite. For all $\sigma_c \in \Sigma_1$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$ is called. We also see that from premise 1 and inspecting the mode checking judgements, the inductive hypothesis is applicable to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. By inductive hypothesis, each call to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$ terminates and there are finite number of such calls. It follows that $\texttt{ips}(\mathcal{L}, j, \pi, \sigma_{\text{in}}, \varphi_1 \wedge \varphi_2)$ terminates.

Case $\varphi \equiv \exists \vec{x}.\varphi$.

According to the definition of $\texttt{ips}$, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists \vec{x}.\varphi) = \texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists \vec{x}.\varphi) \setminus \{\vec{x}\}$. From premise 1 and inspecting mode checking judgements, we see the induction hypothesis is applicable. By inductive hypothesis, $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \exists \vec{x}.\varphi)$ terminates from which it follows that $\texttt{ips}(\mathcal{L}, j, \tau, \pi,$ $\sigma_{in}, \exists \vec{x}.\varphi)$ terminates.

Case $\varphi \equiv \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2)$.

According to the definition of $\texttt{ips}$, to calculate $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \forall \vec{x}.(\varphi_1 \rightarrow \varphi_2))$ we first make a call to $\texttt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$. Let $\Sigma_1 \leftarrow \texttt{ips}(\mathcal{L}, j, \tau, \pi,$

$\sigma_{in}, \varphi_1)$. From premise 1 and inspecting the mode checking judgements, we see that the inductive hypothesis is applicable. By inductive hypothesis, $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \varphi_1)$ terminates. From Lemma 15, we know that $|\Sigma_1|$ is finite. For all $\sigma_c \in \Sigma_1$, a call to $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$ is made. From premise 1 and inspecting the mode checking judgements, we can again see that the inductive hypothesis is applicable to $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$. By inductive hypothesis, each such call to $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_c, \varphi_2)$ terminates and there are finite number of such calls. It follows that $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{in}, \forall \vec{x}.(\varphi_1 \to \varphi_2))$ terminates.

Case $\varphi \equiv \varphi \, \mathcal{S}_{[c,d]} \varphi_2$ and $\mathbf{B} \in \mathit{label}(\varphi)$

Sub-Case To show (1):

Let $\Sigma_1 \leftarrow \pi.\mathcal{A}(\varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{R}$. From the definition of $\mathtt{ips}$, $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2) = \Sigma_{out} = \sigma_{\text{in}} \bowtie \Sigma_1$. From premise (1) and inspecting the mode checking judgements, we see that the inductive hypothesis of Lemma 15 (1) is applicable. By inductive hypothesis, $|\Sigma_1|$ is finite. Thus, the join operation terminates and it follows that $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \sigma_{\text{in}}, \varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)$ terminates.

Sub-Case To show (2):

From definition of $\mathtt{uSS}$, to calculate Σ_β, $\mathtt{ips}(\mathcal{L}, i, \tau, \pi, \bullet, \varphi_2)$ is called once. From premise, Lemma 4, and inspecting applicable mode checking judgements, we see that inductive hypothesis of (1) is applicable. From inductive hypothesis (1), we can say that $\mathtt{ips}(\mathcal{L}, i, \pi, \bullet, \varphi_2)$ terminates.

According to the proof of Lemma 15 (1), we know that $|\Sigma_\beta|$ is finite. We also know that $\pi.\mathcal{A}(\varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)(i-1).\mathbb{S}_\beta$ is finite. From this, we can say calculating $S^{\beta}_{\text{remove}}$ and S^{β}_{new} terminates. Finally, we can say that calculating $\pi.\mathcal{A}(\varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{S}_\beta$ terminates, as each of the set is finite.

While calculating $\pi.\mathcal{A}(\varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{S}_\alpha$, for each $\langle \sigma, k \rangle$ pair in $\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta$, $\mathtt{ips}$ is called once. From premise, Lemma 4, and inspecting applicable mode checking judgements, we see that inductive hypothesis of (1) is applicable. By inductive hypothesis (1), each such call to $\mathtt{ips}$ terminates. There are finite such calls to $\mathtt{ips}$ as there are finite $\langle \sigma, k \rangle$ pairs in $\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta$. Hence, calculating Σ_α terminates. In the same vein, calculating S_{new}, S_{update}, and $\pi.\mathcal{A}(\varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2)(i).\mathbb{S}_\alpha$ terminates.

Finally, in the worst case, while calculating $\pi.\mathcal{A}(\varphi)(i).\mathbb{R}$, each $\langle \sigma, k \rangle$ pairs in $\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta$ is joined with each $\langle \sigma_1, j \rangle$ pairs in $\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha$. As $|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\beta|$ and $|\pi.\mathcal{A}(\varphi)(i).\mathbb{S}_\alpha|$ are finite, the join operations terminate from which it follows that $\mathtt{uSS}$ terminates concluding our proof.

Case $\varphi \equiv \varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2$ and $\mathbf{B} \notin \mathit{label}(\varphi)$

According to the definition of $\mathtt{ips}$, $\mathtt{ips}(\mathcal{L}, j, \tau, \pi, \varphi_1 \, \mathcal{S}_{[c,d]} \varphi_2) = \Sigma_{out} = S_{R_1} \cup S_{R_2}$. It is sufficient to show that the construction of sets S_{R_1} and S_{R_2} terminates. We will then show that $|S_{R_1}|$ and $|S_{R_2}|$ are both finite and thus the set union operation terminates.

We will first show that the construction of S_β terminates. By construction of S_β, a call to $\mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$ is made for all l where $0 \leq l \leq j$ and $c \leq \tau_j - \tau_l \leq d$. From premise 1 and inspecting the mode checking judgements, we see that the inductive hypothesis is applicable to $\mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$. By inductive hypothesis, each call to $\mathtt{ips}(\mathcal{L}, l, \tau, \pi, \sigma_{in}, \varphi_2)$ terminates and there are finite (j+1 in the worst case) number of such calls. It follows that the construction of S_β terminates.

By Lemma 15, $|S_\beta|$ is finite. By construction, each σ is added in S_{R_1} where $\langle\sigma, j\rangle \in S_\beta$ and $c \leq 0 \leq d$. As there are finite number of such σ, the construction of S_{R_1} terminates. By construction, $|S_{R_1}| \leq |S_\beta|$ and it follows that $|S_{R_1}|$ is finite.

By construction of S_{R_2}, for each $\langle\sigma_\beta, k\rangle \in S_\beta$ where $k \neq j$, $\mathbf{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta$
$, \varphi_1)$ is called for all q such that $k < q \leq j$. By inductive hypothesis, each call to $\mathbf{ips}(\mathcal{L}, q, \tau, \pi, \sigma_\beta, \varphi_1)$ terminates and there are finite number of such calls. Thus, the construction of the set S_{R_2} terminates.

As both $|S_{R_1}|$ and $|S_{R_2}|$ are finite, the set union operation terminates.

Thus, it follows that $\mathbf{ips}(\mathcal{L}, j, \tau, \pi, \varphi_1 \,\mathcal{S}_{[c,d]}\varphi_2)$ terminates.

Case $\varphi \equiv \varphi_1 \,\mathcal{U}_{[c,d]}\varphi_2$.

The proof for this case is similar to the preceding case.

□

The following two lemmas state that the **checkCompliance** function is correct and it terminates. The Theorem 1 follows from the following two lemmas.

Lemma 17 (Correctness of **checkCompliance** function)**.** *For all $\mathcal{GMP}$ formulas φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all internal states $\pi = (\mathcal{A}, i)$ where $i \in \mathbb{N}$, for all empty environments η_0, such that: (1) π is strongly consistent at $i \in \mathbb{N}$ with respect to $\mathcal{L}$, τ, and φ, (2) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, (3) $\{\}, \{\} \vdash \varphi : \chi_O$, if **checkCompliance**$(\mathcal{L}, j, \tau, \pi, \varphi) = truthValue$, then $(truthValue = true) \leftrightarrow \exists\sigma.(\mathcal{L}, \tau, j, \eta_0 \models \varphi\sigma)$.*

Proof. The proof follows from the soundness argument of **ips** correctness, Lemma 14. □

Lemma 18 (Termination of **checkCompliance** function)**.** *For all $\mathcal{GMP}$ formula φ, for all $j \in \mathbb{N}$, for all logs $\mathcal{L}$, for all time stamp sequences τ, for all state $\pi = (\mathcal{A}, i)$, for all empty environments η_0, such that: (1) π is strongly consistent at $i \in \mathbb{N}$ with respect to $\mathcal{L}$, τ, and φ, (2) $i \geq j$ and $\tau_i - \tau_j \geq \Delta(\varphi)$, (3) $\{\}, \{\} \vdash \varphi : \chi_O$, the function **checkCompliance**$(\mathcal{L}, j, \tau, \pi, \varphi)$ terminates.*

Proof. The proof follows from the termination argument of the **ips** function (Lemma 16 (1)), as according to the definition of **checkCompliance** function, only the **ips** function is called from the **checkCompliance** function. □

E Policies and their Associated Mode Specification

The HIPAA policy we use in our experiments is shown in Figure 13 and the mode specification for predicates used in this policy is shown in Table 3. The HIPAA policy is we use for our experiments, contains rules from the following clauses of HIPAA: §164.502(a)(1)(i), §164.502(a)(1)(iv), §164.502(g)(3)(ii)(A), §164.510(a), §164.512(b)(1)(v), §164.512(j)(1)(ii)(A), §164.512(j)(1)(ii)(B), §164.508(a)(2), and §164.502(g)(3)(ii)(B). To get the original interpretation of HIPAA for these selected clauses, one can just replace upper bound on the past temporal operators, `bound`, to ∞.

The GLBA policy we use in our experiments is the conjunction of the policies shown in Figure 14 and Figure 15. The mode specification for predicates used in this policy in shown in Table 2. The GLBA policy we use for our experiments, contains rules from the following clauses of GLBA: §6802(a), §6802(b), §6802(d), and §6803(a). To get the original interpretation of GLBA for these selected clauses, one can just replace upper bound on the past temporal operators, `bound`, to ∞.

$$\forall p_1, p_2, q, m, d, u, t.(\mathsf{send}(p_1, p_2, m) \wedge \mathsf{info}(m, d, u) \wedge \mathsf{contains}(m, q, t)) \longrightarrow \Bigg($$

$$\Big((\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{samePerson}(p_2, q) \wedge \mathsf{attrIn}(t, \mathrm{PHI}))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{attrIn}(t, \mathrm{PHI}) \wedge (\exists m_1.((\diamondminus_{[0,\mathrm{bound}]}(\mathsf{send}(q, p_1, m_1))) \wedge \mathsf{isValidAuthz}(m_1, p_1, p_2, q, t, u))))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{attrIn}(t, \mathrm{PHI}) \wedge (\mathsf{purpose}(u, \mathrm{treatment}) \vee \mathsf{purpose}(u, \mathrm{payment})$$
$$\vee\, \mathsf{purpose}(u, \mathrm{healthCareOperations})) \wedge (\exists m_2.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(q, p_1, m_2)) \wedge \mathsf{isValidConsent}(m_2, p_1, p_2, q, t, u))))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge (\mathsf{inrole}(p_2, \mathrm{clergy}) \vee (\mathsf{notin}(t, \mathrm{religiousAffiliation}) \wedge \exists m_3.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(p_2, p_1, m_3))$$
$$\wedge\, \mathsf{isDirectoryRequestByName}(m_3, p_2, p_1, q, t, u)))) \wedge \mathsf{attrIn}(t, \mathrm{directoryInfo}) \wedge \mathsf{purpose}(u, \mathrm{directory})$$
$$\wedge\, (((\forall m_5.\mathsf{send}(q, p_1, m_5) \rightarrow \mathsf{isNotDirectoryObjection}(m_5, p_1, p_2, q, t, u))\ \mathcal{S}_{[0,\mathrm{bound}]}\ (\exists m_4.\mathsf{send}(p_1, q, m_4) \wedge$$
$$\mathsf{isOpportunityToObject}(m_4, p_1, p_2, q, t, u))) \vee (\mathsf{notPracticalToProvideOpportunityToObject}(p_1, p_2, q, t, u) \wedge$$
$$\mathsf{consistentWithPriorPreference}(p_1, p_2, q, t, u) \wedge \mathsf{believesInBestInterest}(p_1, p_2, q, t, u))))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{provider}) \wedge (\mathsf{workForceMemberOf}(p_1, p_2) \vee \mathsf{providerOfMedicalSurveillance}(p_1, p_2) \vee$$
$$\mathsf{providesInjuryEvaluation}(p_1, p_2)) \wedge \mathsf{inrole}(p_2, \mathrm{employer}) \wedge \mathsf{workForceMemberOf}(q, p_2) \wedge$$
$$((\mathsf{attrIn}(t, \mathrm{workPlaceFindings}) \wedge \mathsf{purpose}(u, \mathrm{obligationToRecordWorkPlaceInjury}))$$
$$\vee\, (\mathsf{attrIn}(t, \mathrm{medicalSurveillanceFindings}) \wedge \mathsf{purpose}(u, \mathrm{obligationToPerformWorkPlaceSurveillance}))) \wedge$$
$$\exists m_6.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(p_1, q, m_6)) \wedge \mathrm{isNoticeofWorkplaceDisclosure}(m_6)))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{inrole}(p_2, \mathrm{lawEnforcement}) \wedge \mathsf{attrIn}(t, \mathrm{PHI}) \wedge \mathsf{toIdentifyOrApprehend}(u, q)$$
$$\wedge\, \mathsf{consistentWithAppLaw}(p_1, p_2, q, t, u) \wedge \exists m_7.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(q, p_1, m_7)) \wedge \mathsf{isAdmissionOfCrime}(m_7)$$
$$\wedge\, \mathsf{believesCrimeCausedSeriousHarm}(p_1, m_7)) \wedge \mathsf{notLearnedWhileTreatingPropensityForCrime}(p_1, q, t) \wedge$$
$$\mathsf{notLearnedThroughRequestForTreatment}(p_1, q, t) \wedge (\exists m_8.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(q, p_1, m_8))$$
$$\wedge\, \mathsf{isAdmissionOfCrime2}(m_8, q) \wedge \mathsf{containsMsg}(m, m_8) \wedge \mathsf{contains}(m_8, q, t) \wedge \mathsf{attrIn}(t, \mathrm{attribute\text{-}list\text{-}164.512j3}))))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{inrole}(p_2, \mathrm{lawEnforcement}) \wedge \mathsf{attrIn}(t, \mathrm{PHI}) \wedge \mathsf{toIdentifyOrApprehend}(u, q)$$
$$\wedge\, \mathsf{consistentWithAppLaw}(p_1, p_2, q, t, u) \wedge \mathsf{believesEscapeLawfulCustody}(p_1, q))$$

$$\bigvee$$

$$(\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge (\mathsf{parentOf}(p_2, q) \vee \mathsf{guardianOf}(p_2, q) \vee \mathsf{localParentOf}(p_2, q))$$
$$\wedge\, \mathsf{attrIn}(t, \mathrm{PHI}) \wedge \mathsf{permittedByOtherLaw}(p_1, p_2, q, t, u))\Big)$$

$$\bigwedge$$

$$(\mathsf{notinrole}(p_1, \mathrm{coveredEntity}) \vee \mathsf{notin}(t, \mathrm{psychNotes}) \vee (\exists m_9.((\diamondminus_{[0,\mathrm{bound}]}\mathsf{send}(q, p_1, m_9))$$
$$\wedge\, \mathsf{isValidAuthz}(m_9, p_1, p_2, q, t, u))) \vee (\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{forCounselingOrTrainingPrograms}(u, p_1))$$
$$\vee\, (\mathsf{inrole}(p_1, \mathrm{coveredEntity}) \wedge \mathsf{forDefenseInLegalProceeding}(u, p_1, q)))$$

$$\bigwedge$$

$$(\mathsf{notinrole}(p_1, \mathrm{coveredEntity}) \vee (\mathsf{notParentOf}(p_2, q) \wedge \mathsf{notGuardianOf}(p_2, q) \wedge \mathsf{notLocalParentOf}(p_2, q)) \vee$$
$$\mathsf{notin}(t, \mathrm{PHI}) \vee \mathsf{notProhibitedByOtherLaw}(p_1, p_2, q, t, u))\Bigg)$$

Figure 13: HIPAA policy used in our empirical evaluation.

send (-,-,-)
contains (+,-,-)
info (+,-,-)
isNoticeOfDisclosure (+,+,+,+,+,+)
affiliateOf (-,+)
notconsumerof (+,+)
notin (+,+)
isOptOut (+,+,+,+,+,+)
isNoticeofPotentialDisclosure (+,+,+,+,+,+)
inrole (-,+)
notinrole (+,+)
nonAffiliateOf (+,+)
consumerOf (-,+)
organizationOf (-,+)
attrIn (+,+)
purpose (+,+)
existsConfidentialityAgreement (+,+,+)
lawyerOf (+,-)
notpurpose (+,+)
processConsumerAuthorizedService (+,+)
securitizationSale (+,+)
samePerson (-,+)
extendsCreditOnBehalf (-,+)
maintainConsumerAccount (-,+)
isConsentForDisclosure (+,+,+,+,+,+)
protectRecordSecurity (+,+)
beneficialInterestOf (-,+)
financialRepresentativeOf (-,+)
forResolvingCustomerDispute (+,+)
ratingAgencyOf (-,+)
complianceAssesor (-,+)
attorneyOf (-,+)
accountantOf (-,+)
auditorOf (-,+)
specificallyPermittedOrRequiredByLaw (+,+,+,+,+)
inAccordanceWithRightToFinancialPrivacyActOf1978 (+,+,+,+,+)
subUnitOf (-,+)
forSale (+,+)
forMerger (+,+)
forTransfer (+,+)
forExchange (+,+)
inAccordanceWithFairCreditReportingAgency (+,+,+,+,+)
isConsumerReport (+)
isResponseTo (+,+)
authorizedByLaw (+)
newCustomer (-,+)
renewedCustomer (-,+)
certifiedPublicAccountOfAState (+)
subjectToEthicalDisclosureProvision (+)

Table 2: Mode definition of predicates of the GLBA policy (Figure 14 and Figure 15) used in the experiments.

F Experimental Results

In this section, we present the experimental results for our empirical evaluation. Figure 16 shows the HIPAA experimental result for the average execution time over different trace length for varying bounds when the event traces are stored in a memory-backed database. Figure 17 shows the

send (-,-,-)
contains (+,-,-)
info (+,-,-)
attrIn (+,+)
inrole (-,+)
samePerson (-,+)
isValidAuthorization (+,+,+,+,+,+)
purpose (+,+)
isValidConsent (+,+,+,+,+,+)
notin (+,+)
isDirectoryRequestByName (+,+,+,+,+,+)
notPracticalToProvideOpportunityToObject (+,+,+,+,+)
consistentWithPriorPreference (+,+,+,+,+)
believesInBestInterest (+,+,+,+,+)
isOpportunityToObject (+,+,+,+,+,+)
isNotDirectoryObjection (+,+,+,+,+,+)
workForceMemberOf (-,+)
providerOfMedicalSurveillance (-,+)
providesInjuryEvaluation (-,+)
isNoticeofWorkplaceDisclosure (+)
notinrole (+,+)
notParentOf (+,+)
notGuardianOf (+,+)
notLocalParentOf (+,+)
notProhibitedByOtherLaw (+,+,+,+,+)
toIdentifyOrApprehend (+,+)
consistentWithApplicableLaw (+,+,+,+,+)
isAdmissionOfCrime (+)
believesCrimeCausedSeriousHarm (+,+)
notLearnedWhileTreatingPropensityForCrime (+,+,+)
notLearnedThroughRequestForTreatment (+,+,+)
isAdmissionOfCrime2 (+,+)
containsMsg (+,+)
believesEscapeLawfulCustody (+,+)
permittedByOtherLaw (+,+,+,+,+)
parentOf (-,+)
guardianOf (-,+)
localParentOf (-,+)
forCounselingOrTrainingPrograms (+,+)
forDefenseInLegalProceeding (+,+,+)

Table 3: Mode definition of predicates of the HIPAA policy (Figure 13) used in the experiments.

comparative maximum memory usage (excluding the event trace) of **précis** over **reduce** for the HIPAA experiment just above. Figure 18 shows the HIPAA experimental result for the average execution time over different trace length for varying bounds when the event traces are stored in a disk-backed database. It is very apparent that the relative speed of **précis** increases over **reduce** in case the event trace is stored in a disk-backed database. It is also apparent that with the increasing bounds the memory usage of **précis** for storing the summary strucutres increases significantly faster that **reduce**. When the event trace is stored in a disk-backed database, **précis** achieves a speedup of 3.5x-10x over **reduce** which is higher than the speedup **précis** achieves over **reduce** when traces are stored in a memory-backed database (2.5x-6.5x).

Figure 19 shows the GLBA experimental result for the average execution time over different trace length for varying bounds when the event traces are stored in a memory-backed database. Figure 20 shows the GLBA experimental result for the average execution time over different trace length for varying bounds when the event traces are stored in a disk-backed database. As the number of **`B-formulas`** in the GLBA policy is 4 out of 9 (less than 50%), the speedup achieved by **`précis`** over **`reduce`** is not as significant as in the case of the HIPAA policy. Moreover, the speedup achieved by **`précis`** over **`reduce`** does not vary in the disk-backed and memory-backed cases for the same reason. Figure 21 shows the comparative maximum memory usage (excluding the event trace) of **`précis`** over **`reduce`** for the GLBA experiment where the event traces were stored in a in-memory SQLite3 database. **`précis`**'s memory consumption increases with the increase of the bound, of the past temporal operator, over **`reduce`**. This is to be expected as with the increase of the bound on the past temporal operators, **`précis`** has to store more substitutions in the associated summary structures. The curve representing the maximum memory usage of **`précis`** flattens out when the trace length exceeds the bound because after that **`précis`** has the substitutions for bound number of steps at each point of time.

$$\Bigg(\forall p_1, p_2, q, m, d, u, t.(\mathsf{send}(p_1, p_2, m) \land \mathsf{info}(m, d, u) \land \mathsf{contains}(m, q, t)) \rightarrow$$

$$\Bigg(((\mathsf{notinrole}(p_1, \mathrm{institution}) \lor \mathsf{affiliateOf}(p_2, p_1) \lor \mathsf{notconsumerof}(q, p_1) \lor \mathsf{notin}(t, \mathrm{npi}))$$

$$\lor (\diamondminus_{[0,\mathrm{bound}]}(\exists m_1.((\mathsf{send}(p_1, q, m_1) \lor \exists p_{100}.(\mathsf{send}(p_1, p_{100}, m_1) \land \mathsf{lawyerOf}(p_{100}, q))) \land \mathsf{isNoticeOfDisclosure}(m_1, p_1, p_2, q, t, u))))\lor$$

$$(\Diamond[0, 30](\exists m_2.((\mathsf{send}(p_1, q, m_2) \lor \exists p_{101}.(\mathsf{send}(p_1, p_{101}, m_2) \land \mathsf{lawyerOf}(p_{101}, q))) \land \mathsf{isNoticeOfDisclosure}(m_2, p_1, p_2, q, t, u)))))$$

$$\lor (\mathsf{processConsumerAuthorizedService}(u, q) \lor \mathsf{securitizationSale}(u, q) \lor \exists p_{107}.((\mathsf{samePerson}(p_{107}, p_1)$$

$$\lor \mathsf{extendsCreditOnBehalf}(p_{107}, p_1)) \land (\mathsf{maintainConsumerAccount}(p_{107}, q))))$$

$$\lor (\exists m_6.((\diamondminus_{[0,\mathrm{bound}]}((\mathsf{send}(q, p_1, m_6) \lor (\exists p_{108}.\mathsf{send}(p_{108}, p_1, m_6) \land \mathsf{lawyerOf}(p_{108}, q))) \land \mathsf{isConsentForDisclosure}(m_6, p_1, p_2, q, t, u)))))$$

$$\lor (\mathsf{protectRecordSecurity}(u, q) \lor \mathsf{beneficialInterestOf}(p_2, q) \lor \mathsf{financialRepresentativeOf}(p_2, q)\lor$$

$$\mathsf{purpose}(u, \mathrm{requiredRiskControl}) \lor \mathsf{forResolvingCustomerDispute}(u, q)) \lor (\mathsf{inrole}(p_2, \mathrm{insuranceRateAdvisoryOrg})\lor$$

$$\mathsf{inrole}(p_2, \mathrm{guarantyAgency}) \lor \mathsf{ratingAgencyOf}(p_2, p_1) \lor \mathsf{complianceAssesor}(p_2, p_1) \lor \mathsf{attorneyOf}(p_2, p_1)\lor$$

$$\mathsf{accountantOf}(p_2, p_1) \lor \mathsf{auditorOf}(p_2, p_1)) \lor (\mathsf{specificallyPermittedOrRequiredByLaw}(p_1, p_2, q, t, u)$$

$$\lor \mathsf{inAccordanceWithRightToFinancialPrivacyActOf1978}(p_1, p_2, q, t, u) \lor \mathsf{inrole}(p_2, \mathrm{lawEnforcementAgency})\lor$$

$$\mathsf{inrole}(p_2, \mathrm{selfRegulatoryOrganization}) \lor \mathsf{purpose}(u, \mathrm{publicSafetyInvestigation}))\lor$$

$$((\mathsf{inrole}(p_2, \mathrm{consumerReportingAgency}) \land \mathsf{inAccordanceWithFairCreditReportingAgency}(p_1, p_2, q, t, u))\lor$$

$$(\diamondminus_{[0,\mathrm{bound}]}(\exists p_{111}, m_8.(\mathsf{send}(p_{111}, p_1, m_8) \land \mathsf{inrole}(p_{111}, \mathrm{consumerReportingAgency}) \land \mathsf{isConsumerReport}(m_8)$$

$$\land \mathsf{contains}(m_8, q, t))))) \lor (\exists p_{110}.(\mathsf{subUnitOf}(p_{110}, p_1) \land \mathsf{consumerOf}(q, p_{110}) \land (\mathsf{forSale}(u, p_{110}) \lor \mathsf{forMerger}(u, p_{110})$$

$$\lor \mathsf{forTransfer}(u, p_{110}) \lor \mathsf{forExchange}(u, p_{110})))) \lor (\mathsf{purpose}(u, \mathrm{complianceWithLegalRequirements})$$

$$\lor \mathsf{purpose}(u, \mathrm{complianceWithInvestigation}) \lor \mathsf{purpose}(u, \mathrm{complianceWithSummons}) \lor (\exists m_7.(\diamondminus_{[0,\mathrm{bound}]}(\mathsf{send}(p_2, p_1, m_7)$$

$$\land (\mathsf{inrole}(p_2, \mathrm{judicialProcess}) \lor \mathsf{inrole}(p_2, \mathrm{governmentRegulatoryAuthority})))\land$$

$$\mathsf{isResponseTo}(m, m_7) \land (\mathsf{purpose}(u, \mathrm{examination}) \lor \mathsf{purpose}(u, \mathrm{compliance}) \lor \mathsf{authorizedByLaw}(u))))))\Bigg)$$

$$\bigwedge$$

$$\Bigg(((\mathsf{notinrole}(p_1, \mathrm{institution}) \lor \mathsf{affiliateOf}(p_2, p_1) \lor \mathsf{notconsumerof}(q, p_1) \lor \mathsf{notin}(t, \mathrm{npi})) \lor ((\forall m_3.((\mathsf{send}(q, p_1, m_3)$$

$$\lor \exists p_{104}.(\mathsf{send}(p_{104}, p_1, m_3) \land \mathsf{lawyerOf}(p_{104}, q))) \land \mathsf{isOptOut}(m_3, p_1, p_2, q, t, u)) \rightarrow \mathit{false})\, \mathcal{S}_{[30,\mathrm{bound}]}(\exists m_4.(\mathsf{send}(p_1, q, m_4)$$

$$\lor \exists p_{105}.(\mathsf{send}(p_1, p_{105}, m_4) \land \mathsf{lawyerOf}(p_{105}, q))) \land \mathsf{isNoticeofPotentialDisclosure}(m_4, p_1, p_2, q, t, u))))$$

$$\lor (\mathsf{inrole}(p_1, \mathrm{institution}) \land \mathsf{nonAffiliateOf}(p_2, p_1) \land \mathsf{consumerOf}(q, p_1) \land \mathsf{attrIn}(t, \mathrm{npi}) \land \mathsf{purpose}(u, \mathrm{performServices})$$

$$\land \mathsf{existsConfidentialityAgreement}(p_1, p_2, t) \land (\diamondminus[0, \mathrm{bound}](\exists m_5.(\mathsf{send}(p_1, q, m_5)$$

$$\lor \exists p_{106}.(\mathsf{send}(p_1, p_{106}, m_5) \land \mathsf{lawyerOf}(p_{106}, q))) \land \mathsf{isNoticeofPotentialDisclosure}(m_5, p_1, p_2, q, t, u))))$$

$$\lor (\mathsf{processConsumerAuthorizedService}(u, q) \lor \mathsf{securitizationSale}(u, q) \lor \exists p_{107}.((\mathsf{samePerson}(p_{107}, p_1)$$

$$\lor \mathsf{extendsCreditOnBehalf}(p_{107}, p_1)) \land (\mathsf{maintainConsumerAccount}(p_{107}, q))))$$

$$\lor (\exists m_6.(\diamondminus[0, \mathrm{bound}]((\mathsf{send}(q, p_1, m_6) \lor (\exists p_{108}.\mathsf{send}(p_{108}, p_1, m_6) \land \mathsf{lawyerOf}(p_{108}, q))) \land \mathsf{isConsentForDisclosure}(m_6, p_1, p_2, q, t, u))))$$

$$\lor (\mathsf{protectRecordSecurity}(u, q) \lor \mathsf{beneficialInterestOf}(p_2, q) \lor \mathsf{financialRepresentativeOf}(p_2, q)$$

$$\lor \mathsf{purpose}(u, \mathrm{requiredRiskControl}) \lor \mathsf{forResolvingCustomerDispute}(u, q)) \lor (\mathsf{inrole}(p_2, \mathrm{insuranceRateAdvisoryOrg})$$

$$\lor \mathsf{inrole}(p_2, \mathrm{guarantyAgency}) \lor \mathsf{ratingAgencyOf}(p_2, p_1) \lor \mathsf{complianceAssesor}(p_2, p_1) \lor \mathsf{attorneyOf}(p_2, p_1)\lor$$

$$\mathsf{accountantOf}(p_2, p_1) \lor \mathsf{auditorOf}(p_2, p_1)) \lor (\mathsf{specificallyPermittedOrRequiredByLaw}(p_1, p_2, q, t, u)$$

$$\lor \mathsf{inAccordanceWithRightToFinancialPrivacyActOf1978}(p_1, p_2, q, t, u) \lor \mathsf{inrole}(p_2, \mathrm{lawEnforcementAgency})\lor$$

$$\mathsf{inrole}(p_2, \mathrm{selfRegulatoryOrganization}) \lor \mathsf{purpose}(u, \mathrm{publicSafetyInvestigation})) \lor ((\mathsf{inrole}(p_2, \mathrm{consumerReportingAgency})$$

$$\land \mathsf{inAccordanceWithFairCreditReportingAgency}(p_1, p_2, q, t, u)) \lor (\diamondminus_{[0,\mathrm{bound}]}(\exists p_{111}, m_8.(\mathsf{send}(p_{111}, p_1, m_8)$$

$$\land \mathsf{inrole}(p_{111}, \mathrm{consumerReportingAgency}) \land \mathsf{isConsumerReport}(m_8) \land \mathsf{contains}(m_8, q, t)))))\lor$$

$$(\exists p_{110}.(\mathsf{subUnitOf}(p_{110}, p_1) \land \mathsf{consumerOf}(q, p_{110}) \land (\mathsf{forSale}(u, p_{110}) \lor \mathsf{forMerger}(u, p_{110}) \lor \mathsf{forTransfer}(u, p_{110})\lor$$

$$\mathsf{forExchange}(u, p_{110})))) \lor (\mathsf{purpose}(u, \mathrm{complianceWithLegalRequirements}) \lor \mathsf{purpose}(u, \mathrm{complianceWithInvestigation})$$

$$\lor \mathsf{purpose}(u, \mathrm{complianceWithSummons}) \lor (\exists m_7.(\diamondminus_{[0,\mathrm{bound}]}(\mathsf{send}(p_2, p_1, m_7) \land (\mathsf{inrole}(p_2, \mathrm{judicialProcess})$$

$$\lor \mathsf{inrole}(p_2, \mathrm{governmentRegulatoryAuthority}))) \land \mathsf{isResponseTo}(m, m_7) \land (\mathsf{purpose}(u, \mathrm{examination})$$

$$\lor \mathsf{purpose}(u, \mathrm{compliance}) \lor \mathsf{authorizedByLaw}(u))))))\Bigg)$$

$$\bigwedge$$

$$\Bigg(((\mathsf{notinrole}(p_1, \mathrm{institution}) \lor \mathsf{affiliateOf}(p_2, p_1) \lor \mathsf{notconsumerof}(q, p_1) \lor \mathsf{notin}(t, \mathrm{accountNumber}) \lor \mathsf{notpurpose}(u, \mathrm{marketing}))\lor$$

$$(\mathsf{inrole}(p_1, \mathrm{institution}) \land \mathsf{inrole}(p_2, \mathrm{consumerReportingAgency}) \land \mathsf{consumerOf}(q, p_1) \land \mathsf{attrIn}(t, \mathrm{accountNumber})))\Bigg)\Bigg)$$

Figure 14: GLBA policy (conjunct-1) used in our empirical evaluation. The rules in this policy correspond to the privacy rules §6802(a), §6802(b), and §6802(d) of GLBA.

$$
\begin{gathered}
\Bigg(\forall p_i, q_i.(\mathsf{inrole}(p_i, \text{institution}) \wedge (\mathsf{newCustomer}(q_i, p_i) \vee \mathsf{renewedCustomer}(q_i, p_i))) \rightarrow \\
(((\exists m_9.(\Diamond_{[0,365]}(\mathsf{send}(p_i, q_i, m_9) \vee (\exists p_{112}.(\mathsf{send}(p_i, p_{112}, m_9) \wedge \mathsf{lawyerOf}(p_{112}, q_i)))\wedge \\
(\mathsf{contains}(m_9, p_i, \text{npiPoliciesAndPractices}) \vee \mathsf{contains}(m_9, p_i, \text{npiCategoriesCollected})\vee \\
\mathsf{contains}(m_9, p_i, \text{npiSecurityPolicies}) \vee \mathsf{contains}(m_9, p_i, \text{npiDisclosuresToAffiliates}))))) \vee (\\
\mathsf{certifiedPublicAccountOfAState}(p_i) \wedge \mathsf{subjectToEthicalDisclosureProvision}(p_i))))\Bigg)
\end{gathered}
$$

Figure 15: GLBA policy (conjunct-2) used in our empirical evaluation. The rule in this policy correspond to the privacy clause §6803(a) of GLBA.

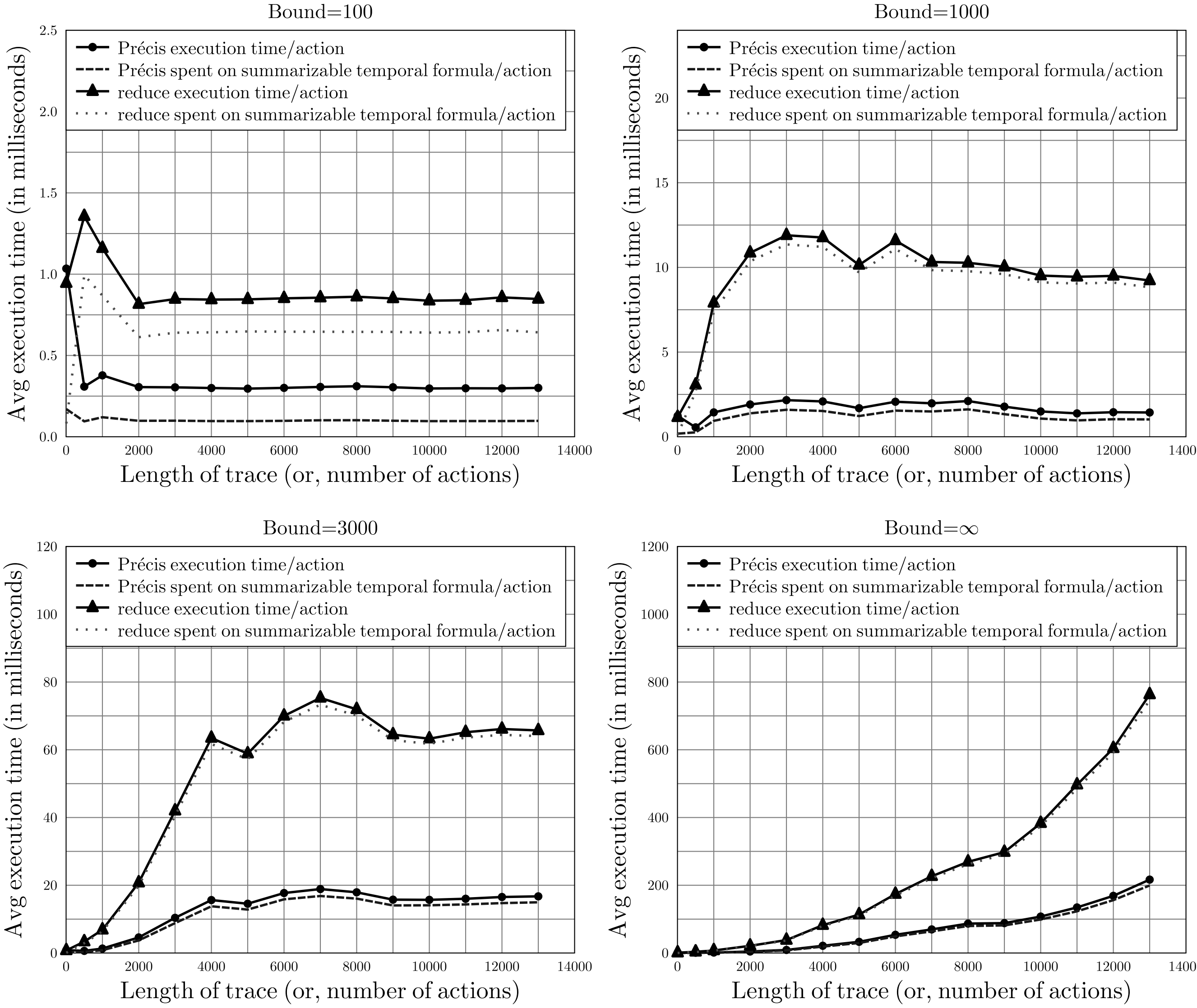


Figure 16: Experimental timing results (HIPAA) with memory-backed database

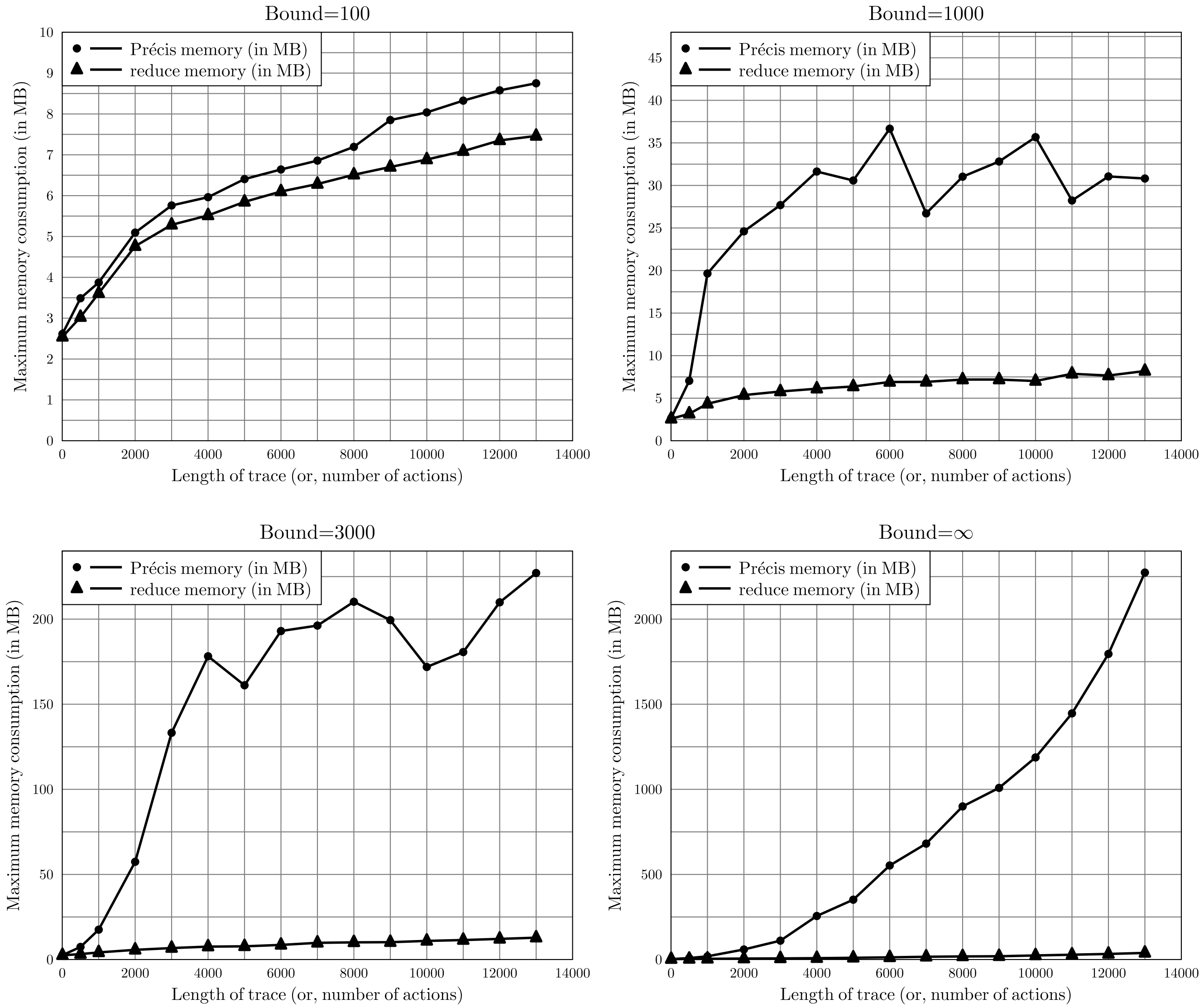


Figure 17: Experimental memory results (HIPAA) with memory-backed database

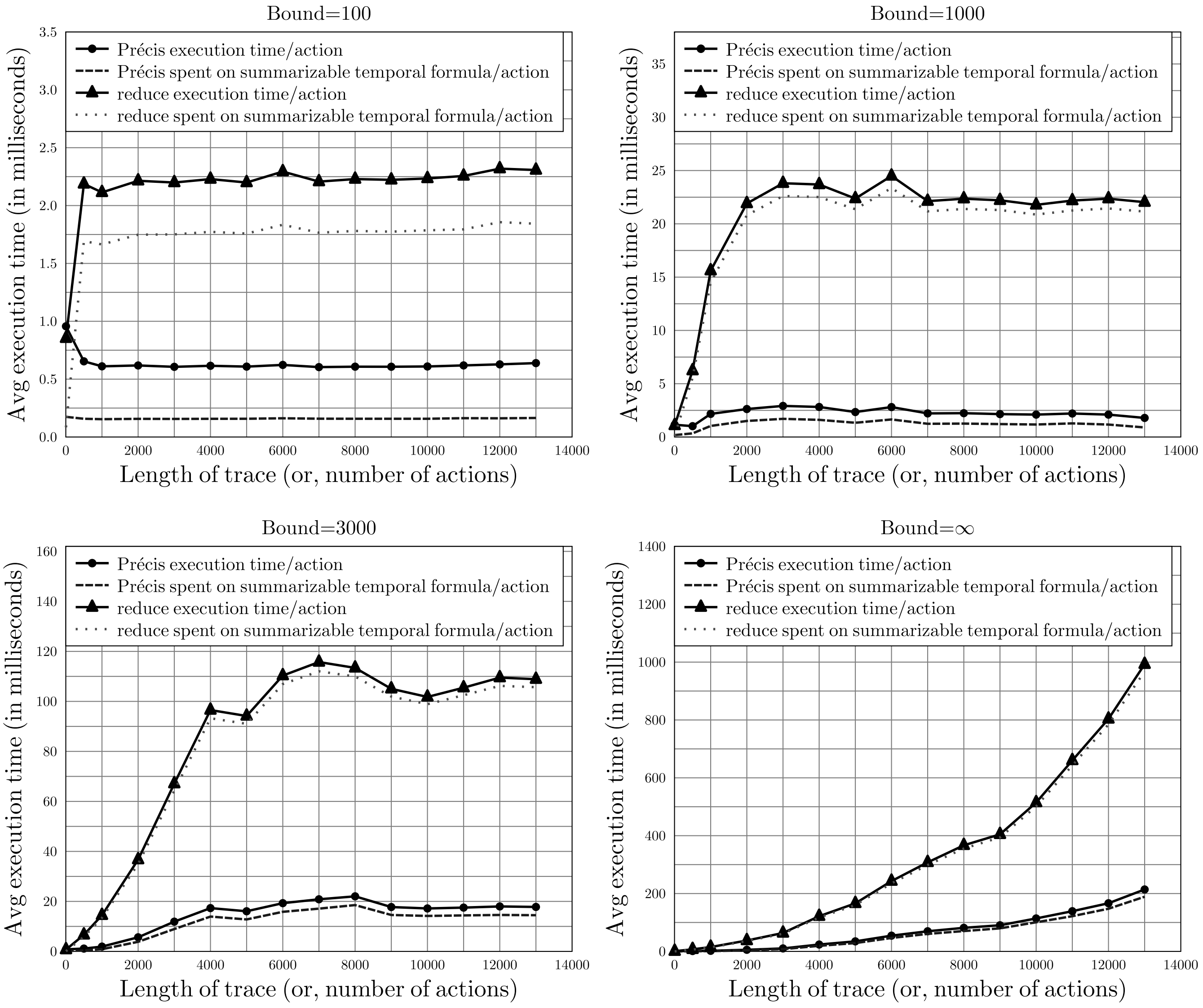
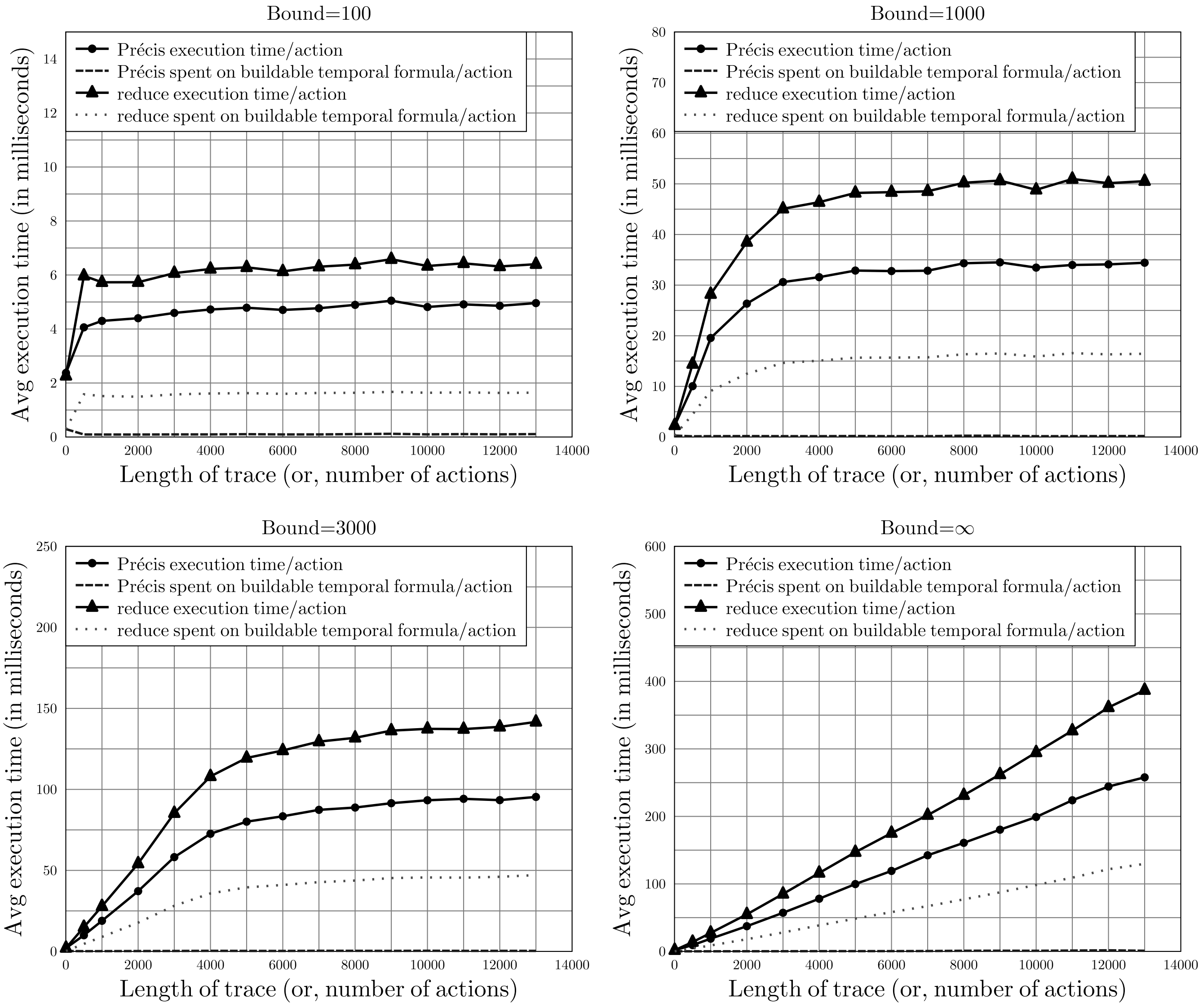


Figure 18: Experimental timing results (HIPAA) with disk-backed database


Bound=100

Bound=1000

Bound=3000

Bound=∞

Figure 19: Experimental timing results (GLBA) with memory-backed database

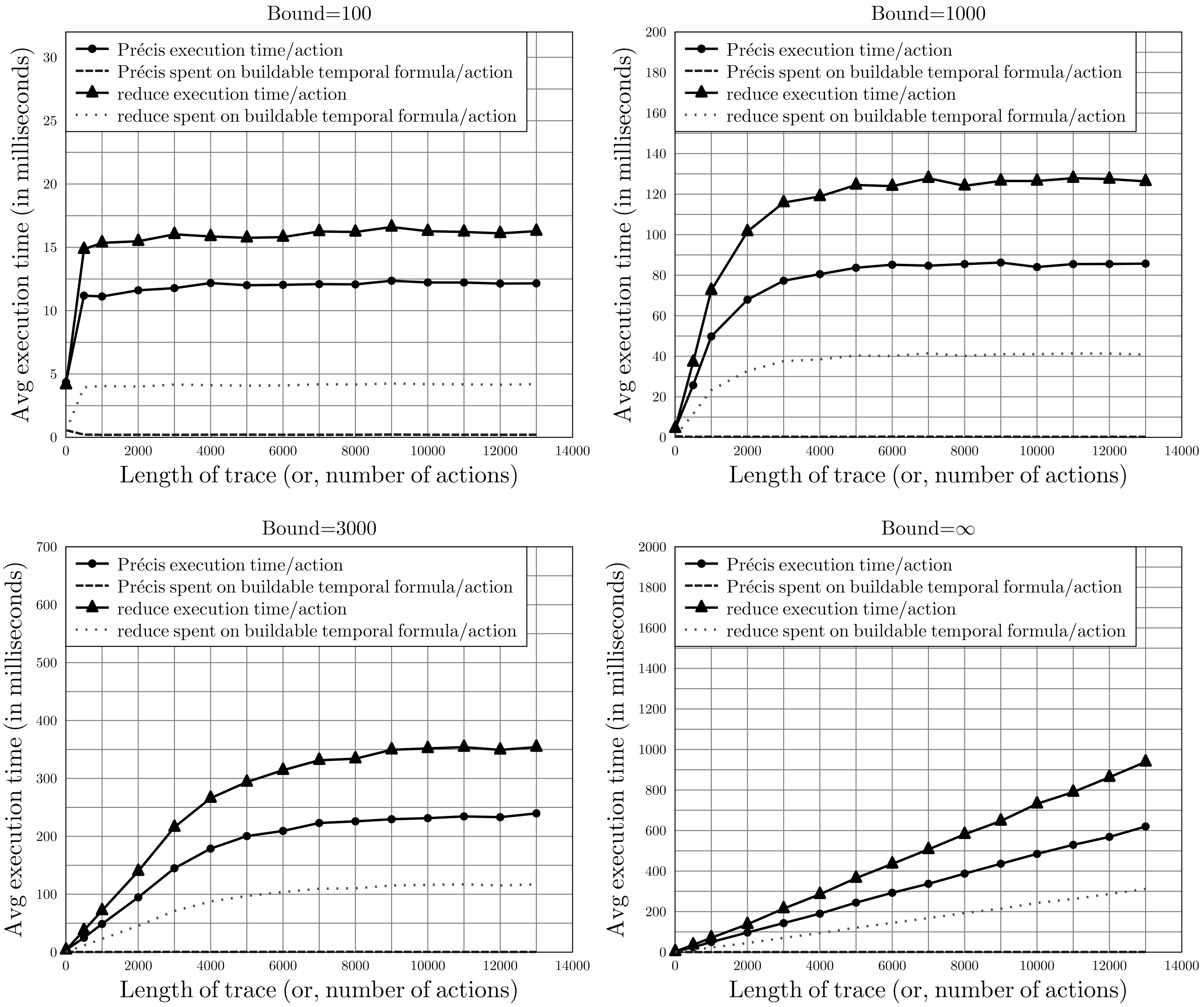


Figure 20: Experimental timing results (GLBA) with disk-backed database

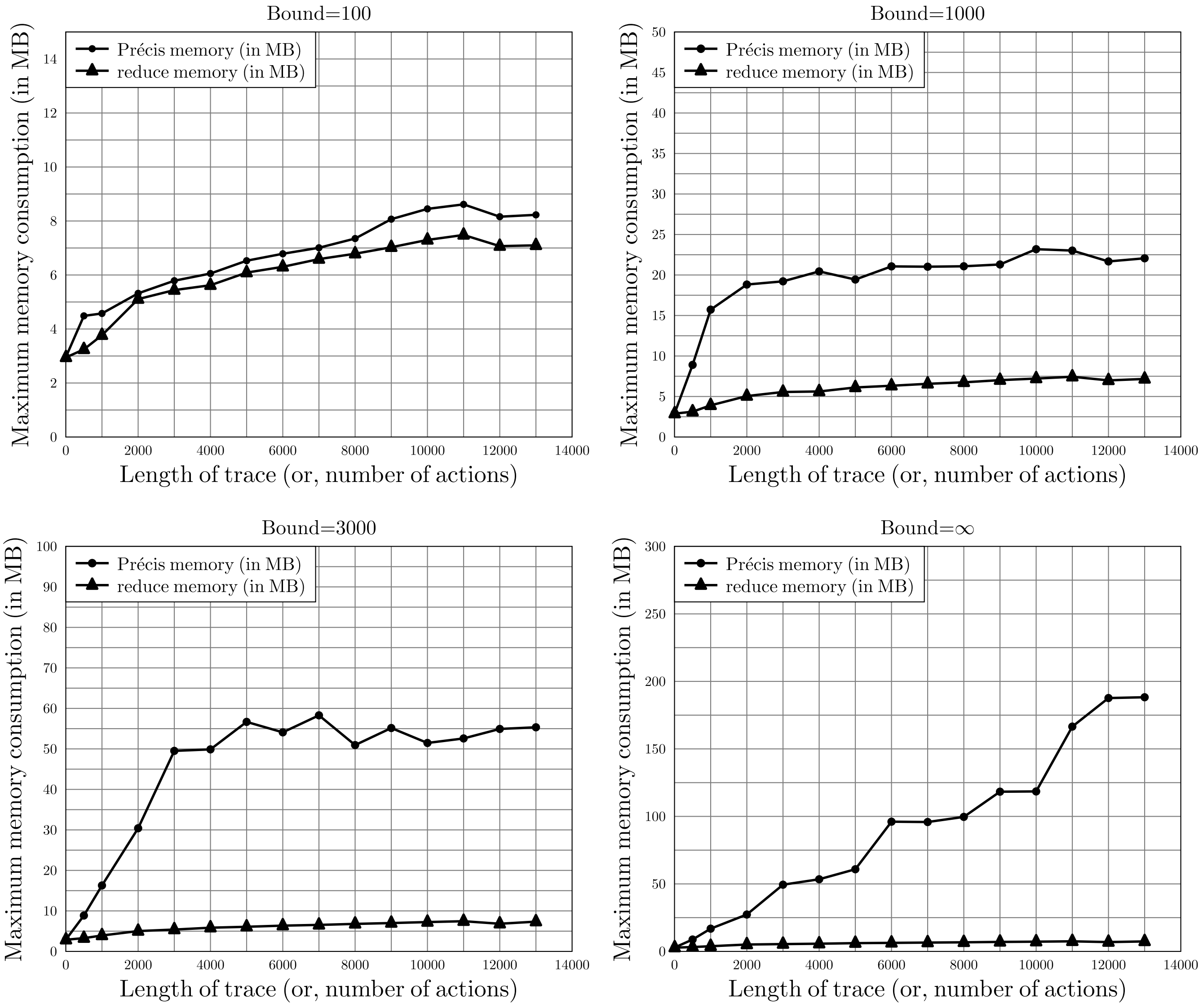


Figure 21: Experimental memory results (GLBA) with memory-backed database